UNIVERSITY OF CALIFORNIA
SANTA CRUZ

PROCEDURAL, PLAYER-CENTRIC GAME BALANCING

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTATIONAL MEDIA

by

Samuel Shields

March 2026

The Dissertation of Samuel Shields
is approved:

Dr. Edward F. Melcer, Chair

Dr. Michael Mateas

Dr. Adam Smith

Dr. Luke Dicken

Peter F. Biehl
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Procedural, Player-Centric Game Balancing

by

Samuel Shields

Game balance is a term widely used among players, researchers, and designers of games. It is a concept that feels vitally important to how we make and play games, but when we try to define it or implement it, we seldom get the same definition twice. Balance appears differently to whoever is judging it, but as researchers and designers, we still must translate this element of game design into technical practice. It is also an expensive and time-consuming subject, one that requires a constant loop of playtesting and design iteration through nearly the entirety of the game development process.

This work seeks to focus our understanding of balance while offering procedural methods to increase speed or improve quality when performing balancing tasks in game design and research. It accomplishes this by offering a taxonomy of balance along with a generic design framework that can be used to apply balancing strategies in any game context. In addition, it provides a catalog of balancing methods, allowing designers to use common patterns to apply procedural balancing to their games. Finally, I offer three technical examples using the taxonomy and framework, putting theoretical knowledge of balance into concrete technical systems.

Balance ultimately helps us design games that make us feel *fairness* in our play. By sharpening, optimizing, and improving our understanding of the term, we improve the games we make and open new doors in game system design.

To Sasha,

For helping me see the beauty in the world.

Acknowledgments

I am overflowing with gratitude for those in my life who supported me throughout this doctorate and I deeply believe that one needs to share their thanks when they feel it. First and foremost, I owe all of my work to my partner Sasha and my loyal dog Smiley. You both have brought light into my life in every way imaginable. You gave me the courage and hope to pursue my passions and dreams in video games and supported me through all the trials and tribulations I faced while pursuing those goals. The importance of a hug, a wagging tail, and a quiet seat in front of the ocean has been demonstrated to me time and again through you two. I have a feeling that my work and creativity will continue to be nurtured by being near both of you.

Also to my family; who introduced me to games so young that I cannot remember a time without them. I have a distinct memory of receiving a copy of *A Link to the Past* on my birthday. Getting lost in Hyrule, seeing the intricacies of dungeon design, and being able to have an adventure from the living-room CRT—these were formative memories for me and inspire me to make games that will do the same to some other curious child. This is to say nothing of the constant curiosity, immersion in fantasy and science fiction, and respect for the outdoors that you instilled in me. I hope I have made you proud. The fact that I will be the first Dr. in our entire known family history blows my mind.

Next up is Eddie—you have been the best mentor I could have asked for. I wasn't always the easiest (most focused, agreeable, etc.) student, but you persisted and believed in me through it all. You taught me how to manage criticism, sort meaning-

ful and meaningless feedback, celebrate wins, and use losses to supercharge the next attempt. Your work inspired me to the point where I wanted to follow (probably too) closely in embodied design before I pivoted to game balancing. You pushed me to follow the fire of my passion, identify and cut scope, all while giving me so many opportunities in research and writing. Above all, you are a good friend and an excellent person, and I can't wait to see where your research takes you next.

Here is where I have to condense down a little bit, as I think this section might be longer than the dissertation if I give everyone a separate paragraph.

To the UCSC faculty, in particular Noah Wardrip-Fruin, Michael Mateas, Adam Smith, Elin Carstensdottir, Michael John, Sri Kurniawan, and Adam Sporka. You all influenced me greatly at different points in my dissertation, directly or through your historical body of work. The lineage of students behind you speaks to the quality of your leadership and insights. To Luke Dicken and Rob Zubek: my time at Zynga was formative, and seeing games in action in a real environment was priceless. I love seeing how you all bring our academic experiments to industry. To my co-authors and mentees, Eddie, Ollie Withington, Ross, Alex, Michael, Noah, Celine, Shi, Danny, and Ramon, collaborating with you all was an honor, and I am fortunate to exchange ideas with you all. The entirety of the UCSC Computational Media cohort is amazing, and I would grab a coffee with any of them at any time. I especially want to thank the members of the Alternative Learning Technologies lab and the Expressive Intelligence Studio. The folks in both groups proved instrumental to my work.

There are a litany of people outside of UCSC and my family who inspire me along the way. In terms of researchers, Mike Cook, Gillian Smith, Alex Jaffe, Max

Kreminski, Jasmine Otto, Barrett Anderson, Julian Togelius, Tanya Short—all of you played a major role in forming my academic and intellectual identity. In terms of game designers, Masahiro Sakurai, Mark Rosewater, Brian Bucklew, and Hideo Kojima all come to mind as core inspirations. Athletes also inspire me, especially as it comes to Rock Climbing and Skateboarding—Tommy Caldwell, Miho Nonaka, Sasha DiGiulian, Nyjah Huston all come to mind. My game and support groups (and dear friends) also play a core role—my magic pods in the bay (Ben, Sawyer, Brian, Mia, Eduardo, Bjarke), my DND party (Vlad, Erik, Raj, Frank, Sukanya, Devesh), my volunteer and support friends (Neil, Neal, Brian, Derrick, Rob, Rich, Steve, among others). Particular shout outs to my close friends Johnathan Pagnutti (who played a key role in inspiring me to finish this work), Kyle Cochran, Nick Duckwiler, Johnny Dasteel. The Movement/-Planet Granite gym staff, along with all National Park workers, also thank you, as my escapades into the mountains kept me sane through many trials and tribulations.

I needed to have a lot of personal growth to do this—intellectually, emotionally, and spiritually. I am going to do the sappy thing and drop a bunch of lessons I found the most helpful throughout this process. Always keep a beginner's mindset. Staying curious about other people and their expertise helps both in growing your own knowledge and in understanding the perspectives of others. Love and friendship are fundamental to everything. Take it one day at a time. Resentments and paranoia are self-administered poison, trust people first. The answer to most questions worth asking is nearly always "it depends". Spend time learning how to separate constructive and destructive feedback and protect your soul from those who wish to hurt you. Always assume positive intention in others. If you have something good to say about someone, share it immediately.

Recognize and respect your own limitations. Build a community, stand by them and watch what grows from the coming together of people. Take the cotton out of your ears and stick it in your mouth. Progress, not perfection. Every topic has unimaginable depth and will reveal itself if you keep prodding it. Rest when you need it. If you are in a pinch, know that it too will pass. What will be, will be.

Looking ahead, I have one goal that I want to follow through; a goal that has been with me since the start of this whole journey. I want to make things that people play. That's it. Play has given me the world and a life worth living—I want to do the same for others. Taking my expertise and making new styles of play, wherever that happens, is my next step.

Preface

I present the work done throughout my degree in a conceptual order—it is meant to have you thoroughly learn the definition, theory, and design tactics behind game balancing before introducing you to explicit technical exercises that prove out the theory. This separates out the dissertation into four sections—a primer on balance, a deep dive into balancing definitions and strategy, an accounting of my technical work, and a look ahead and what work is possible in the future regarding game balancing.

Research, however, hardly ever proceeds along such clean lines. My work formed a more sinusoidal path of triumphs and failures, and my research involves a conversation between my practical, technical work and the theory that informs it. As such, there is a different reading order that might suit you well if you prefer a more chronological accounting of my work. That's right, you can take Deleuze and Guattari's recommendations and read this darned thing out of order! The reading order (of chapters) would be 1, 8, 2, 3, 4, 9, 5, 7, 10, 6, 11, 12. This inserts the technical experimentation where they occurred during theory development.

For a brief summary of the story: I started off my degree bright-eyed and bushy-tailed regarding the potential of automatic game design and its potential applications to the platform fighter genre. Working with Ross Mawhorter, we found technical success in designing an (if I say so myself) impressive evolutionary system with automated playtesting that seemed to generate balanced games... at least, according to our fitness function, which was optimized for competitive play. When I brought it out to players, you can imagine my horror that my positive quantitative metrics did not at all

match the qualitative feedback from the players regarding game balance. Discouraged at first, I had to search for answers on why this occurred, and settled on the fact that I must have taken balance at surface level and it deserved a more nuanced take before I returned to technical efforts to test the concept.

This is where I spent a *long* time reading papers on game balance. I quickly picked up that balance was something that happened in the player and not the game—it is a reaction to gameplay, not an inherent property of the game system itself. It took a lot more data crunching and re-reads of paper after paper to realize that balance was this beautiful, multi-faceted jewel of a concept. It meant a lot of things to different people, but it really was a term used to discuss the *fairness* of a game.

With my newfound understanding of balance, I returned to the IDE, building out another project that aimed to investigate balancing for different sub-definitions of the term. I wanted to see a single architecture cater to multiple personas of players, while leveraging procedural techniques to enable dynamic balancing during runtime. Building another complicated simulation testbed, I found success in using an AI Director pattern to tune for different types of player-agents. Throughout this design process, the combination of applied work and the swirling of paper-codes in my head led to the development of a framework that could be applied regardless of balancing concept, which formed design principles I could follow and recommend to other designers and researchers.

It's here that I returned to evaluation—I was dissatisfied with a simple summary of stats from my last experiment. I met Oliver Withington at GDC 2025, who gave an excellent talk on Expressive Range Analysis. After many awkward-time-zone

zoom meetings, we landed on the curious question—what if we applied ERA to game balancing? Could an ERA plot show us the diversity and expressive nature of different game balancing techniques? This launched my final stage of research and experimentation, where I aimed to create tooling that gave a more "narrative" oriented view of game balancing. If balancing is about fairness, and fairness waxes and wanes throughout gameplay, then game balance tells a story anchored by our feelings and experience of fairness. As a fan of horror, this immediately described why I loved the genre so much—we move back and forth into extremes of unfairness (I'm low on ammo, health, and a crazy monster is behind me) and fairness (I'm in a safe room, nothing can get me, I can finally save). This stage of work saw me flesh out evaluation strategies, not just in their current forms but in a new system that aims to elaborate on the ways we can see game balance.

Towards the end of the degree, I wanted to create guides for researchers and designers to keep taking the work forward. I developed a catalog of targets and techniques so that future systems designers had established conceptual patterns to engage with. I then reflected on future opportunities and all the things I wish I could have researched. I stuck those projects towards the end of this work, and I hope I plant the seeds of curiosity in some researcher down the line.

Research is hardly clean and organized, and I hope this preface gives you a more human look at what my process looks like—a rise-fall-rise narrative arc that wrestled between experimentation and theory. Now buckle in and enjoy the rest of the dissertation!

Part I

What Even *is* Game Balance?

Chapter 1

Introduction

"...the perception of balance is more powerful than balance itself" [231]

Jeff Kaplan, Lead Designer of *Overwatch* [41]

Game balance is a term widely used among players, researchers, designers, and commentators of games [33, 137]. It is often used casually to describe some essential quality of games. Epithets such as "that is so overpowered", "I was over-leveled by the end" or "everyone is using the same strat" are common in games where there is discordance in game balance. When balance is lacking, player discontent inevitably follows [48, 16]. Industry practitioners recognize this; balance is recognized as key to both general player engagement and enjoyment, as well as long-term retainment of players in a game's ecosystem. The urgency and importance of balance means that it appears at every step of the game development life cycle—nascent during the design loop, critical during iterative playtesting, and an overwhelming focus during post-release patch and

content releases. Despite this, it is fairly (pun intended) unclear what exactly balance is, how to implement it, or how to evaluate it once the game becomes playable. Despite its widespread usage, balance seems to mean something very different depending on the context in which it is used. How can such a term be used to describe the competitive, multi-player environments of eSports while also seemingly applying to casual players in a whimsical adventure game? Moreover, the strategies for implementing balance also land all over the map—games like *Left 4 Dead* [498] have runtime systems to shift balance during gameplay, but fighting games like *Street Fighter* [66] have static implementations that are meant to stay put throughout the game's lifecycle. Going further, how we evaluate if a game is balanced also seems ill-defined and diverse. Saying *Magic: The Gathering* [395] is a balanced game can simply be evaluated by looking at the win-rates of tournaments and distributions of dominant decks[1]; yet designers also frequently discuss the density of interactions between players and the aesthetics of game elements in discussions on the game's balance. How can all the facets of game balance seemingly be in tension with each other?

A brief anecdote to put these tensions more concretely: When I was a precocious pre-teen, the third-space of our neighborhood was wherever there was a *Nintendo Gamecube*, four controllers, and a copy of *Super Smash Bros. Melee* [173]. It was a default mode for socializing. Bored? Smash Bros. Something to settle? Smash Bros. Sibling rivalry? Smash Bros. New friend? Smash Bros. The frenetic mix of character expressivity, fluidity of movement, and the surprises of items and levels made it an endlessly entertaining and replayable game to revisit over and over again. After

[1]See `https://magic.gg/events/pro-tour-lorwyn-eclipsed` for an example of statistics on a *Magic* tournament in early 2026

seemingly thousands of hours of play, a sub-group of the neighborhood kids became increasingly competitive with the game. Preferred characters were selected, complex strategies enacted, deep mechanics such as 'wavedashing' were uncovered and honed to give advantages over those who did not practice such dark arts. We insisted on wired controllers and CRT televisions to mitigate any possible input and feedback lag that could show up in the game. In other words (myself being a member of said group), we got a bit... *sweaty*. As we developed our mastery, we became less and less welcome with the rest of the neighborhood kids who played the game. If we showed up, the disk was often swapped out for something more equitable, such as *Mario Party* [196]. As such, we started migrating to game stores and local tournaments, where we were met with similar minds and strategies, and the level of challenge significantly ratcheted upward.

After a few years of this, a new title in the franchise was announced with great anticipation—*Super Smash Bros. Brawl* [457]. Neighborhood and tournament players alike became excited, as an expanded roster and set of levels meant opportunities for an even greater diversity of fun and chaos. However, when it was finally released, it was received differently among the audiences in profound ways. It slowed down gameplay, making combos, advanced techniques, and reactive gameplay difficult or impossible. It introduced "tripping", which would randomly stun your character during movement, providing an opponent ample opportunity to punish even when the player ostensibly made no mistake in the first place. In tournament scenes, the game seemed to punish and discourage pure skill. Within weeks, game stores and competitions reverted to *Melee* as the game of choice. Back in the neighborhood, the game was received with gusto. Suddenly, our competitive sect of players were welcomed back into the neighborhood,

as everyone felt they had a chance against even our practiced mastery.

I argue that despite neither of these games fulfilling the needs of both the more casual neighborhood or the tournament scene, they both are successful examples of balance in game design, just for different *player contexts*. *Melee* rewarded mastery and de-emphasized randomness, while *Brawl* encouraged chaos and chance so that all players had some hope of participation. Even in the same genre, the same game series, the balance of these games was simultaneously divergent *and* successful. Returning to our discussion of balance—how can these opposing examples both be called game balance and both be successful in their endeavors?

This dissertation aims to tackle the tensions of game balance by situating it as a multidimensional term with a wide array of implementation strategies and evaluation techniques that are highly dependent on a desired player audience, designer goals, and interpretation of fairness in games. It provides guidance on how to identify what type of balance your game needs, designer strategies to proceduralize and implement this balance, and tooling suggestions to quickly evaluate if your game is hitting the right equilibrium for your balance goals. I also provide experimental and empirical evidence of these processes through the implementation and testing of procedural balancing systems (*BrawlerAGD* and *FighterDDA*). In terms of contributions, I provide ways to carve out your own subsection of game balance, making the term useful for research and design, where too narrow or broad a definition might cloud or invalidate a game or experiment's interpretation of balance. I then provide not only manual, established ways to balance a game, but also catalog and suggest methods to proceduralize balancing operations. As balance occurs throughout nearly every part of game development,

automating even a few percentage points of the task means a drastic saving in time, playtesting resources, and ultimately the overall cost of game development [422, 548]. Considering procedural methods to balancing is also increasingly important when PCG and automatic game design (AGD) are increasingly common in new games. If a designer is hands-off during the creation of content, they necessarily need to be hands-off when balancing said content, making procedural game balancing critical to any procedural system. Finally, I provide suggestions for the evaluation of if a game's balance is in, well, balance. Balance is an emergent, hard-to-describe trait of games that requires clever techniques to detect and evaluate—tooling, player modeling, and analytics techniques all play a crucial role in helping a designer or researcher know if a game or game system is hitting the mark. Taken together, this dissertation provides a framework, sampling of tooling, and experimental implementations of that framework to help others focus, optimize, and evaluate game balancing for *any* game that needs it.

A brief road map for what is ahead[2]: In Chapter 2, I highlight the multidimensionality of the term "game balance", dive into why it is so important, and talk more explicitly about the foundational pieces of the term (audience, design goal, methods, game element targets, and evaluation). In Chapter 3, I provide a theoretical infrastructure for understanding balance. This includes highlighting how balance is an emergent, phenomenological term at heart, that balance is deeply tied to fairness, and how we can consider fairness from both top-down and bottom-up contexts during our attempts to define balance. In Chapter 4, I provide a taxonomic definition of balance that high-

[2]While this dissertation is organized *conceptually*, the *chronological narrative* of the work occurs in a slightly different order. Details on the chronological story and how you can read the dissertation to respect the chronology can be found in the preface.

lights its varied definitions while providing designers, researchers, and players valuable sub-categorizations of the term that can be applied to their game of interest. Chapter 5 presents a generic framework for game balancing, which allows the usage of taxonomic concepts from the previous chapter in a holistic pattern that allows the game balancing design process to be conceptualized and executed throughout the implementation and evaluation of a game. Chapter 6 provides a catalog of balancing strategies, providing templates on how to execute and automate game balancing. Chapter 7 follows this by providing evaluation strategies for balancing, helping provide the lens through which one can understand if a game is balanced or not. After this, the focus of this dissertation switches to focusing on experimental implementations of the game balancing framework presented in previous chapters. Chapter 8 looks at *BrawlerAGD*, a system meant to design balanced fighting games. Chapter 9 reviews *FighterDDA*, a game that seeks to tune game balance at runtime for a variety of player personas. Chapter 10 covers the tooling used for analyzing the expressivity of playtraces generated during the balancing process—*Playtrace Arc Search*, or PAS. The last part of this dissertation looks ahead, suggesting opportunities for future work (Chapter 11) and a summary of the conclusions of the dissertation as a whole (Chapter 12).

Pull out your weighing-scales, your tight-ropes, your laser-levels: we are about to dive into game balance from every angle we have available to us.

Chapter 2

Background

> "...abstractly I think balance is a collection of tuning parameters that are tweaked and modified in a certain way to provide a particular experience for a particular audience."
>
> Brie Chin-Deyerle, Lead Engineer at *343 Industries* and *Riot Games*

It is not the case that game balancing is underrepresented in games academic (professional, community) discourse. Quite the opposite—there are a plethora of approaches to defining, implementing, and evaluating balance across nearly every game context you could imagine. In a literature review I conducted (presented in detail in Chapter 4), reviewing the academic literature alone over the last 25 years yielded 95 publications focused on the subject. At the time of this dissertation's publication, a casual search of "game balance" in the Game Developer Conference's archive of talks

yields hundreds of results [208], and likewise GameDeveloper.com will yield *thousands* of results for the same term [207]. A visit to r/gamedev (a community dedicated to game design with over 2 million members) will yield an overwhelming number of posts on the subject [384]. It is apparent that those who make and play games have a decent investment in game balance and readily discuss it.

In addition to established literature on the subject, I have also conducted interviews with designers and contributors from AAA, Indie, and Community Modding projects. They have been gracious enough to provide a thorough account of their experience in game balancing on game series such as *Arc Raiders* [119], *Halo* [56], *Spyro* [143], *Rivals of Aether* [99], *Sparrow Warfare* [335], *Metroid* [382] and *Final Fantasy* [459] randomizers, among other projects. The rest of this dissertation will pull quotes from these interviews, helping to connect academic and example projects with the larger picture of how publicly available games are balanced. A complete list of interviewees, their background, and interview protocols can be found in Appendix A.

When we try to define game balance and look into the world for guidance, there seems to be an implied, shared knowledge about what it is—but can we define what that knowledge is concretely? Which audience is using it in the most accurate way? How do we even begin to define balance?

2.1 How Would You Define Balance?

To start, we can break up the audiences discussed in the prior section into three main groups: academics, designers, and players. Looking at their explicit and

implicit definitions of what game balance is should inform what game balance is and understand why it can be used so universally in so many different contexts.

2.1.1 Academia

Game balance research has a large footprint in academia, reflecting a deep interest in the concept and how it is applied to research and design [546]. Game balance has been researched in the context of procedural systems [409], metagames [186], user experience [318], dynamic difficulty adjustment [26], parameter tuning [287], etc. However, despite the range and depth of literature on game balance, the actual definitions and discussion of what game balance is tend to be inconsistent depending on the domain and author. Many academic publications adopt a singular and explicit definition of balance. For example, some more well known definitions include meeting the right level of challenge [73] or two players of equal skill having equal win rates [101] or ensuring that no gameplay option is under- or over-powered [217].

More recently, several scholars have begun to view balance as multidimensional, including several more singular definitions, typically with two to four different underlying concepts being outlined. For example, Becker and Görlich [33] breaks balance down into roughly four aspects: "...characteristics of well-balanced games, meaningful decision options, player skill, and the prevention of dominant strategies". Their analysis included 14 publications from academic and practitioner sources and used semantic analysis, noting that, although there were overlapping concepts, sources generally had different understandings of the term. Meanwhile, Pfau and Seif El-Nasr [369] settled on four categories in the context of the MMORPG *Guild Wars 2* [21]—viability, symmetry,

fairness, and difficulty. Another paper considers balance from the perspective of tuning to include players of diverse skill levels, stating "the trick is to stay behind", seemingly contradicting work that focuses on more competition- or difficulty-oriented visions of balance [157].

This all is to say—researchers are deeply interested in game balancing (and how to define it), but there is seldom overlap in how it is defined or researched. The most common interpretations of the term tend to reference equal win-rates for equally skilled players [31, 486], the sense of challenge that is not too boring or difficult [26, 30, 262], and the ability for differentially skilled players to compete with another [503, 150, 29].

2.1.2 Industry

In the game industry, balance is regarded as a complex priority that is key to the longevity and success of games. For instance, Owen Mahoney, the CEO of *Nexon* , which publishes long-running, internationally available games such as *MapleStory* [526] cites balance as the key element in the success and persistence of a game [82]. Similarly, to maintain a large player base and a healthy competitive scene, *Riot Games* publishes balance patches at a frequent cadence to the hit game *League of Legends* [400] to ensure that there are viable options for players of all skill levels while avoiding exploitable strategies in high-level competitive play [247].

While balancing is viewed as important, what balance means to a game and how it is implemented within industry is nuanced and varied. In part, this is due to balance being viewed as difficult to implement because it is difficult to perceive. To quote a *Game Developer* guide on understanding balance: "...there's no strict, defined

line at which a game goes from being in balance to out of balance, it's a gradual continuum" [60]. To this end, game balance has a reputation for being quantitatively driven while relying on feedback from qualitative playtesting for evaluation and refinement. For example, the balancing of *Fallout: New Vegas* [346] began with a spreadsheet of values, but was quickly abandoned in favor of "vibes"-based balancing [381]. Identifying player audiences is also important to a well-tuned game. Michael John (former creative director at EA) said during the development of *Spyro* [143] "...three kids in particular met my three archetypes... [name 1 redacted] was very persistent, but they were a bad game player, and remained a bad game player for years... [name 2 redacted] was my total median player. They got everything in a reasonable amount of time.. [name 3 redacted] was your speedrunner. They finished everything so fast. Made no mistakes."

Practitioner literature aimed at focusing and improving game design and development also highlights the importance and nuanced complexity of game balance. Ralph Koster's *A Theory of Fun for Game Design* takes a light-hearted approach to game creation and cites balancing as "...providing a good level of challenge for the user" [246]. In *Game Design Workshop*, Tracy Fullerton writes a chapter on balance, acknowledging the scope of the subject by claiming that their analysis is a "...vast simplification of what you will actually experience when you try this yourself." Fullerton describes balance as "...the process of making a game better" and offers a specific set of definitions of what they mean by balance:

> Balancing a game is the process of making sure the game meets the goals you have set for the player experience: that the system is of the scope and complexity you envisioned and that the elements of that system are working together without undesired results. In multiplayer games, it means that the starting positions and play are fair (i.e., no player has an inherent advan-

> tage), and no single strategy dominates all others. In single player games, it means that the skill level is properly adjusted to the target audience. For short, we call these four balancing areas variables, dynamics, starting conditions, and skill. [138]

Recognizing the varieties of balance is a pattern in the practitioner literature. Jesse Schell identifies a much wider set of types for balance, while also indicating that their list is not exhaustive. Examples from their types include: Fairness, Challenge vs. Success, Meaningful Choices, Skill vs. Chance, Competition vs. Cooperation, Short vs. Long, Rewards, Punishment. Schell highlights the importance of creating specific problem statements before evaluating balance, using the above categories to test hypotheses [420]. Schreiber and Romero [422] published a design guide dedicated to game balancing that highlights that balance is an emergent property of games. They note that balance is "...first and foremost, a feeling, and you know when you feel it." They note that balance is a gray area that is contextually sensitive, highlighting seven variations of game balance. An important thing to note about these practitioner guides is that they tend to include many definitions and focus on setting goals and narrow hypotheses as an antecedent to executing balance itself.

Interestingly, seemingly contradicting the views above, states of imbalance are also valued in game development. For example, the *BFG* weapon in *Doom (2016)* [204] was intentionally overpowered and labeled a "kill everything" button, presumably to introduce a sense of power in the player [380]. Likewise, the balancing of *Elden Ring* [136] is heavily stacked against the player, but that stacking is precisely the reason why many play and treasure the game [53]. Masahiro Sakurai of the *Smash Bros.* series [27] states that too much balance can make the game feel stale and that wide variability is critical

to keeping a game fun, even if it introduces some occasional states of imbalance [158]. This tension between the desire for balance and the recognition of the importance of imbalance highlights that industry perspectives also involve a diverse range of implementation strategies and definitions.

2.1.3 Players

The conversations between players on a given game's balance take place all over the internet. Just as academics and industry sources talk about balance in a variety of ways, players seem to use balance in many different contexts and use cases. To illustrate some of these discussions and their diversity, I will be providing some references to various game subreddits demonstrating how players discuss and frame game balance with one another, even when the games and how they are played are wildly different. Specifically, I will pull examples from r/SSBM (*Super Smash Bros. Melee* [173]) and r/AnimalCrossing (*Animal Crossing* [341]). Screenshots highlighting discussions concerning the differences between these games and their balance can be found in figure 2.1.

To illustrate—the r/SSBM subreddit has over 110,000 members[1] discussing the state of *Super Smash Bros. Melee*, a heavily-played fighting game that has had patches for over two decades. Despite the game's lack of changes over the years, discussions of balance are still lively and varied. There are discussions on how the game should be balanced now, analysis of millions of online games for balance, and if stages should be banned based on recent progression in expert play. In a vastly different context,

[1] Recorded on August 20th, 2025. Archival link available in appendix B.1.

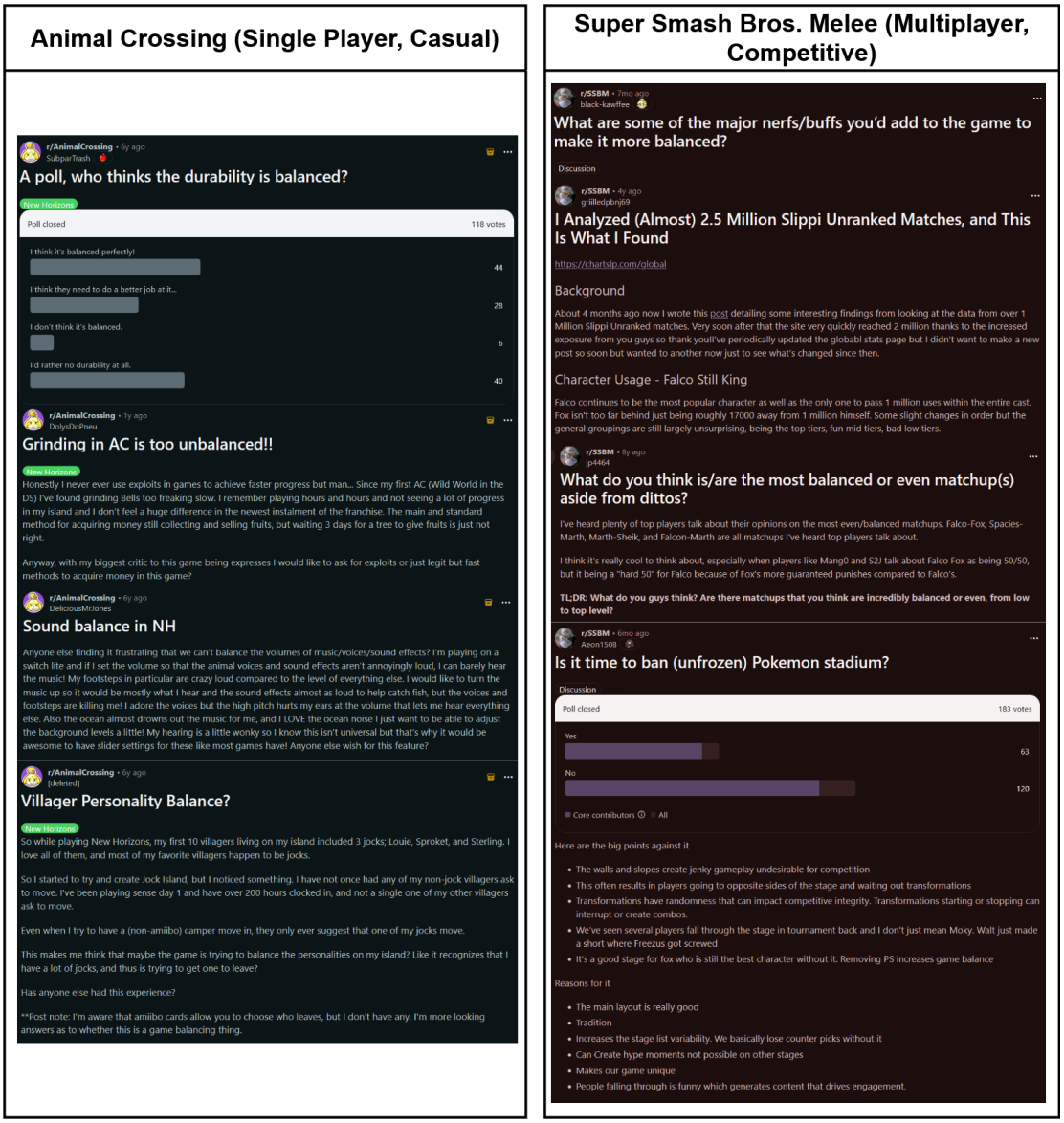

Figure 2.1: A series of screenshots from *Melee* and *Animal Crossing* subreddits demonstrating commentary on the games and varied interpretations of balance. Competitive games like *Melee* orient themselves around character viability and bans, while *Animal Crossing* discussions focus on progression speed, narrative, and aesthetic tuning. Links to these posts can be found in appendix B.1.

the r/AnimalCrossing subreddit with 2.7 million members[2] also hosts discussions on balance. *Animal Crossing*, a game known for its casual, cozy, mostly single-player pace, appears to be the antithesis of the style of play of r/SSBM members. However, discussions on balance are still prominent, though the content of such conversations differ greatly. *Animal Crossing* players post about the progression of resource acquisition, the narrative differences in the appearance of personality dispositions in their residents, and the role of sound is the aesthetic balance of the game.

Reflecting the diversity of opinions on game balancing in academia and industry, players also seem to readily use the word for many game and player contexts, sometimes talking past one another when communicating on the subject.

2.1.4 Trick Question

So, how do we actually define balance? As you might have noticed from the section header—this is a bit of a trick question. Importantly, despite the multidimensional approaches to balance, this spread of terms between authors are still varied definitions that often do not align with one another. It is not the case that any of these single- or multidimensional definitions of game balance are wrong, but rather that their variance suggests a much wider breadth of the concept that could be useful to researchers and designers. Consolidating these different views on game balance into a single taxonomy that accounts for them could help focus the conversation around different styles and goals of game balancing. Such a taxonomy would enable more targeted research to be performed and would allow designers to more readily identify what styles of balance

[2]Recorded on August 29th, 2025. Archival link in appendix B.1.

they are seeking in their games. Instead of attempting to synthesize a taxonomy out of the blue here, I will leave an in-depth discussion on a balance taxonomy in Chapter 4. That taxonomy will be drawn from a literature review, helping to build the foundation of a multi-dimensional version of balance in a way that captures the nuance and texture of the term and its design implications.

Another thing to pull from the above discussion is that, regardless of context or persona, people tend to hint at some implicit sense of *fairness* in their statements. Is a strategy too dominant so that I feel forced to use an option or hopeless in the option I choose? Can I play against my older sibling and still have a chance of winning? Do I feel like the game is too punishing and I need to put it down in frustration? Does it take too long to grind out the basic components of the game so I can get to the good stuff? All of these hint at a kernel of a core critical judgment: "is this a fair experience?".

For now, until we dive into the weeds of a multidimensional definition in Chapter 4, let us work with definition: **Balance is a multidimensional design concept relating to fairness that depends on the players, designers, and game context that is being examined.**[3]

2.2 Why Should We Care About Balance?

With all the fuss around defining balance, we can acknowledge that it is top of mind for many players, designers, and researchers. But why? What makes game balance such a critical element of game design that so many acknowledge and focus on? How long does balancing last, when does it happen, and who is involved?

[3]If you are very eager, you can read the refined definition in section 4.3.

2.2.1 Balance as a Success Criteria

Regarding motivation to make balance the core of design, industry professionals of many sizes and fiscal models (AAA, independent, free-to-play) note balance as critical to their game's sales, player appreciation, and long-term engagement [82, 247, 8]. In an Ubisoft earnings call, it was noted that maintaining balance in existing titles was more profitable than simply releasing sequels [425]. The free-to-play space particularly highlights the importance of balancing, noting that proper balancing can increase spending among consumers [280, 239, 289, 501]. There is a dark side to this, however. Certain balancing strategies might lead to more exploitive practices seen in various free-to-play environments. See chapter 11 for a deeper dive into this subject. Beyond the core success of titles, a properly balanced game can be the seed for thriving eSports and secondary economies surrounding the game. Perhaps the earliest and most foundational example of this is the competitive *Starcraft* [40] scene that creates the foundation for South Korea's general eSports scene to thrive [188]. That being said, one need not look far for examples of games that directly or indirectly fostered massive communities around competitive applications of their balanced ecosystem. Games can even have their development and balancing passed off to the community, forming lively modding, speedrunning, and randomizer communities that add to the history and eminence of any given title. The development of *Get to Work* [213] highlights this by directly looping in speedrunners in marketing, playtesting, and development of their game [214].

It also plays a key role in the identity and playability of a given game. On decisions to move difficulty towards the hard end on *Elden Ring* [136], lead designer

Hidetaka Miyazaki notes that the game is so successful with a specific, mastery-oriented audience because they made the increased balance of challenge a core part of the games identity—noting that the game would not be unique and well-received without such a tuning [293]. Proper balancing also enables players to make interesting decisions. Sid Meier (a prominent 4X/strategy game designer involved in the creation of the *Civilization* [307] franchise) discussed this in a GDC talk highlighting his design process, noting that unbalanced, dominant strategies restrict the amount of interesting and unique actions a player can take [308].

2.2.2 Balancing is Important—and Costly

To answer the question on the implementation costs of balancing, it lasts a long time [231], is a component of design through most of the game development process [221, 245], and involves many different playtesters and designers [240]. It occurs in nearly all stages of the game development lifecycle. When discussing *Halo*, Brie Chin-Deyerle was asked when balancing occurs, responding that "...[balancing all the way to post-release] definitely does... It's later, it's not the first thing... I think the first thing in terms of difficulty tuning is focusing on one difficulty. Getting one difficulty good is hard enough." Going further, Brie mentioned that the tooling developed during the balancing of bots led to studio-wide improvements in speed and quality. Even the sound team used the automated playtest implemented to tune and test how equalization and sound effects were implemented.

Even after alpha and final releases, studios cite balancing as a critical step in keeping their audience engaged. Supergiant games (of *Hades 2* [468] fame) note in patch

notes and interviews that early access builds with frequent updates create a feedback loop with players where their feedback and data are used to fine-tune future challenge and narrative [234, 159]. Riot games has a two-week balancing and patching lifecycle, where a data-analytics-driven approach identifies characters who are over/underpowered across a wide range of player skill levels. This cadence is maintained to try and make sure that every possible character is viable for every player [466, 265]. Blizzard has famously reworked many characters in their *Overwatch* [41] franchise, noting that retuning a character during balancing can take six months or more [231].

It is not just length—it is the pure volume of people involved in the balancing process that makes it tough to grapple with. Riot has a team dedicated to analytics and balancing, with members often having deep expertise and high ranking in their games to inform their design evaluation [399]. Even if they cannot strictly be employed, indie studios often lean on alpha playtesting and privileged discord communities to collect data on balancing and tuning accordingly. Mega Crit Studios, the developer of *Slay the Spire* [306], noted that while they had a development team of just two, their online playtesting presence was large enough to collect balancing data needed to confirm the game was working as intended through extensive data-driven balancing approaches [153]. During the development of *Destiny 2* [57], Bungie uses a "Sandbox" team dedicated to internally balance weapons and abilities before collaborating with other teams to identify edge cases and outliers [327]. To balance a game, you at a minimum need designers and developers to scope and implement balancing changes, and playtesters or a playtesting apparatus to confirm that those changes worked. If the balancing changes impact downstream changes, you may need to involve other teams (new sound effects,

animations, etc.) to accomplish the balancing goals at hand.

2.3 Procedurality in Game Design

A caveat to everything that follows in this section: it is a fool's errand to try and completely balance a game without playtesting and the collection of player opinions. As we shall see in Chapter 3, game balance is an emergent, phenomenological experience at its heart. My work does not seek to fully automate playtesting out of the process, but rather to proceduralize segments of the balancing process such that some percentage of design and development can be trimmed off of the total game balancing endeavor. In the previous section, we covered the sheer cost and importance of game balancing—if even 5-10% of that time or staff is saved, that would lead to massive savings in cost. Even if time were not saved, if the procedural systems improve the quality of game balancing by a similar margin, then the game has that much more of a chance to be successful in the short- and long-term.

That being said, procedural content generation and mixed-initiative co-creative tools demonstrate that automating different elements of the game design process can yield games with greater velocity, higher quality, or novel mechanics and systems. Although randomization and on-demand procedural content generation in games goes back to titles such as *Elite* [46], there was an impetus in the mid-2000s to shift to more component-based and procedural architectures for games to help solve the ballooning costs of development [128]. Togelius et al. [484] did some early work on search-based strategies to procedural content generation, attempting not only to automate content

creation, but find *good* examples of content generated. Smith [454] provides an overview on procedural content generation, providing a wide array of methods that can be used (optimization, grammar-based, constructivist, etc.) to produce player-interactive systems (e.g. levels, quests). While Smith highlights the potential for new player experiences, the creation of offline and online process, they also mention the important consideration of how long/difficult it will be to implement the procedural system in the first place.

This makes PCG a double-edged sword: it can give us diverse, interesting, and high volumes of content; but it can also become an authoring burden in and of itself. In the context of game balancing, we should take this as a signal that *simpler* systems that are "good enough" are the types of system we should be looking for when considering procedurality.

Also going against some traditions in PCG, we noted above that having a human-in-the-loop through playtesting is critical to pulling off the balance process successfully. From this, we can look towards mixed-initiative co-creative systems for guidance on how we can incorporate player evaluations and gameplay into our balancing design process. Early tools such as *Sentient Sketchbook* were proposed in an effort to improve both the speed and quality of iteration cycles of developers [277]. *Tanagra* similarly looked to have human designers identify key points of modification while the computer handled more laborious or obvious tasks that would have wasted the designer's time. In both cases, we see systems that aim to handle some creative workload that may easily be automated or supercharged with artificial intelligence while keeping the ultimate design direction in the hands of the designer. Applying this framing to balance,

we also want to seek solutions in procedural balancing that aim to solve the laborious elements of balancing while leaving the designer-playtester back-and-forth conversation intact.

How might balancing be used in the context of PCG or MI-CC? Simply put—balance heavily involves the tuning of parameters, game mechanics, and aesthetics to drive differences in the emergent experience of playing of a game. However, the combinatoric decision space of tuning a large set of parameters is dauntingly large in size and can be difficult to grapple with both in identifying the right parameters to tune and to what values to tune them to. Consider *Magic: The Gathering*. There are more than 30,000 unique game pieces, all of which have some measure of interaction with one another. Each game piece has a volume of possible attributes (name, type, keywords, mana value, etc.). When designing a new set of cards, how do you possibly digest that massive decision space that is many orders of magnitude more complex than a deck of poker cards? When asked, designers will list heuristics they use to focus design decisions—following cost curves, interaction with different groups of mechanically similar cards, avoiding printing interactions and mechanics known to be "broken", following established aesthetic and narrative patterns, utilizing known power-to-cost patterns, among other things. Using this metaphor, we can use PCG or MI-CC to proceduralize some of these heuristics to more quickly limit the immense search problem of mechanic and parameter tuning when approaching a game.

Anticipating a critique, one might argue—isn't it a stretch to consider game balancing as procedural content? While there might have once been a more strict delineation between "traditional" and "procedural" game design, that barrier is fading. For

example, Cook [88] discusses the reality that the interaction and outcomes of interlocking game design elements mean that game design is effectively *is* game design. I would argue that the diversity and character of the playtraces generated by the tuning of a system itself is a procedural artifact that can be analyzed through the lens of procedural game design [437]. This point will be explored more in chapter 10. In short, how we tune systems to interact end up dramatically impacting the way gameplay feels and is perceived by the player. If this tuning drives diverse sets of experiences with varying character, we too should consider them as procedural content systems that have a character and expressive range themselves. In the context of balance, this means that how a player experiences fairness is a mode of storytelling with a rich tapestry of techniques tied to it.

A final point relating to procedural systems and balance in the context of PCG that aims to generate entire levels, game elements. These systems aim to have massive potential interactive outputs, but can be plagued by their openness. The "1000 bowls of oatmeal" issue describes PCG systems that generate sets of content that are far too similar (and thus bland) [87]. On the other hand, systems might generate output without drama or broken entirely. Researchers have used the musical metaphor of "beats" in different game contexts, referring to how play can be interpreted as rhythms or points of inflection [456, 303]. You might view balance as it relates to PCG as a specific application of "Experience-Driven Procedural Content Generation", where observations of the affect of players determine the operation of a procedural system [535]. That being said—how do we evaluate for that rhythm? Formalizing balancing frameworks gives us a potential outlet; by understanding what makes the game "fair" or in the

sweet spot of challenge for our players, we get evaluation and tuning criteria that can be algorithmically integrated to help solve the problem of selecting quality output from procedural systems.

2.4 Taring the Scale

This background work highlights three key points that I would like to address in this work:

1. **Balance is valuable and meaningful to the game design process and an indicator of a game's success.**
2. **Balance is multidimensional, related to fairness, and difficult to evaluate.**
3. **Balancing creates a massive search problem that can be partially optimized using PCG and MI-CC methods.**

In the following chapters, I will show that the proper scoping of balance through definitions and frameworks combined with procedural methods to guide and aid designers in their balancing pursuits can move the needle in terms of effort and quality regarding a game's balance. If we properly find our niche of balance through audience, design goals, and game context, we eliminate a huge chunk of our ideal balancing search space. Concretizing the remaining problem in terms of procedural systems gets us even closer. With the right designer intuition and playtesting strategy, we should land closer to the ideal balance we want for a game.

Chapter 3

Experiencing Balance

> "My definition is a game is balanced when I'm clenching my jaw, but just enough that it's still feeling good. Not, you know, cracking teeth or anything like that."
>
> Charlene Putney, Co-Founder of NEON AURELIUS and former Larian Studios writer

When I started my research into game balance, I went with my prior assumptions about the term—that *games* possessed balance. If you design a game the right way, it will be objectively balanced, and that is a good thing that is readily appreciable by players, designers, and researchers. When I got my hands dirty with designing[1], however, I found out that balance is not inherent to a game at all—it is not a concrete

[1] See chapter 8 on BrawlerAGD.

topic, it is not a static attribute of the cartridge you stick into your *Nintendo 64*. Balance was something my players *felt*, and everyone felt it in a different way. With this in mind, I had to go back to the drawing board.

What I discovered formed a theoretical pair of axioms that underlies the entirety of this dissertation:

1. **Game Balance is Emergent.**

2. **Game Balance is Ultimately a Phenomenological Judgment.**

These rules are important reminders because they tell us that:

First, we don't get simple controls over how we tune balance. Balance is inherently invisible on its face, the result of low-level components creating complex and unpredictable outcomes at a higher level. Speaking as someone with a passion for system design, it would be lovely if we had a rotary encoder that we could simply turn when we wanted more or less balance. Alas, this is not the case. Balance emerges from the way that different game elements and game systems interact with each other based on the attributes we encode into those elements and systems. We might be able static analysis to get closer to understanding what an emergent outcome of balance might be, but it is only when the car is assembled and running do we understand how it handles on the road.

Second, measuring balance will always end at some form of human judgment, because it is trying to speak to some form of human experience. Communicating and predicting this experience will be fraught with potential for misinterpretation. We

can share elements of that experience and attempt to empathize with another person's perspective, but we will never get a direct look into what another person's experience is. As such, while balance is frequently discussed (particularly in the academic literature) as something we can quantify in its end state, if it is *ultimately* something happening in the privacy of someone's experience, we will need qualitative reporting to understand how our tuning efforts have played out.

If you buy these claims, you can likely skip most of this chapter. If not, in the following sections I will discuss emergence in games and its role in game balance, what it means to view game design through a phenomenological lens, and what we can take from our understanding of "fairness" when considering emergence and phenomenology in game balance design[2].

3.1 Emergence in Games

Emergence can be quickly understood using the common aphorism "the whole is more than the sum of its parts", which is a phrase commonly attributed to Aristotle [405]. Thinkers in the nineteenth century (including John Stuart Mill and G. H. Lewes) explicated the concept, discussing how the interaction of individual linear causal elements can result in phenomena with attributes that cannot be understood by examining the causal elements independently [358]. While the constituent parts of a system can be well-defined, follow rules, and easily measurable, the presence of multiple elements within a system produces an overall whole that can be unpredictable, complex,

[2]This chapter is based on the DiGRA 2024 publication "Towards Qualia-Driven Design"—see Shields et al. [435]

and surprising when put in the context of its founding elements.

Games are a perfect example of how emergence in games is produced at multiple levels of meaning. Most games involve a myriad of inter-connected systems, gearing together and tweaking one another such that unpredictable and novel outcomes are produced for the player. An early and easily digestible example of a game-like emergence lies in John Conway's *Game of Life*, where cellular automata following simple rules can create emergent patterns involving movement, extinction, growth, and so on [145]. For a more modern example, in *The Legend of Zelda: Tears of the Kingdom*, a sandbox environment consisting of game-economy, player stats, game physics, enemy aesthetics, and a building system all combine to make a game that produces extremely varied and dynamic experiences for every play session. The player might spy a camp of enemies below them, including a threatening pseudo-boss NPC prowling around the center of the settlement. The player normally does not know that the difficulty and spawns of these enemies are tuned based on the player's progression through the game, so that the encounter meets the mastery of the player. The player has a limited inventory of tools for building physics-toys (vehicles, weapons), and must choose what types of objects they could build to best distract, defeat, or escape the encampment. Perhaps there is a treasure chest in the center of the enemies; dangling an incentive for the player to engage and potentially flesh out their inventory for future encounters. Taken together, the player is making complex judgments about the enemy difficulty, the physics of the environment, the economics of their inventory and potential rewards—even though these systems taken independently could be grasped relatively easily. Such emergence is the kernel of surprising gameplay, or gameplay that can meet the needs and expectations

of the player.

For the purposes of this work, we will focus primarily on the emergence in the game as defined by Penny Sweetser in their textbook *Emergence in Games* [471]. They define emergence as "...the properties, behaviors, and structure that occur at higher levels of a system, which are not present or predictable at lower levels." They specify that local emergence is the behavior in localized sections of the system, while global emergence occurs in the collective behavior of the system as a whole. For the purposes of game balance, we might view local emergence as the independent systems tuned within a game that end up interacting with others. For example, the complicated evolution of an economy, using drains and faucets to pace the player's rewards and inventory throughout gameplay. Global emergence, on the other hand, can be viewed as the sum of interacting tuned systems resulting in a player experience. Sweetser points out that there are many game elements and systems that are ripe for emergence: game worlds, characters, agents, narrative, and meta-social—all of which should be noted to be common places to tune and evaluate balance. In a way, systems have their emergence enabled by the tuning of elements of their composition so that they are in balance—Conway's *Game of Life*'s emergence depends on the number of neighbors for each cell defined within the system's rules—tuned upward or downward, behavior will change, or the emergence of the system will disappear entirely. Likewise, if we tune the economy to be too forgiving and exploitable (or grindy and punishing), the emergence of complex player strategies to navigate that game economy might dematerialize and cease to be an engaging system entirely.

Emergence also suggests the existence of inputs and outputs that can be ob-

served in any system, with the terminal output being the player experience. When we consider this pipeline for balancing, we see the inputs of an emergent system as the parameters and composing elements that are available for tuning and modification. In terms of outputs, we can directly observe metrics of lower-level local emergent systems that result from tuning during gameplay. However, when we reach that terminal, aesthetic realm of player experience, the meaning of our quantitative measurements can get scrambled or fail to report what is important or successful about the gameplay systems being evaluated. We still have some measure of quantitative data as reported by the game (length of time played, number of actions, and so on) but we can only gesture at what those data mean in terms of player experience if we don't also collect qualitative perspectives. Thus, at different levels of tuning balance for different levels of emergence, we have different procedural and evaluation strategies we have to take—as balance is ultimately a first-person, phenomenological experience.

3.2 Phenomenology in Interactive Design

Before we dive into balance and player experience, let us take a moment to discuss phenomenology. Phenomenology is concerned with understanding the "...conscious experience as experienced from the subjective or first person..." of an individual [453]. Phenomenology was initially formally introduced as an academic discipline in the early twentieth century by Edmund Husserl, who was frustrated that intellectual analysis was increasingly being performed in theoretical terms abstracted from human experience. Husserl opens the field by distinguishing objects of perception and the act of percep-

tion as separate, allowing future philosophers to explore the space between experienced objects and their mental representations.

Husserl had a cadre of acolytes that provided critical dimensions to phenomenology. Heidegger zooms out from perception and includes being, maintaining that perception and representation are cognitive elements of a more encompassing state of being. Schutz incorporated social structures into phenomenology, discussing how experience is informed and reformed by compounding social observations and performances. Heidegger and Schutz are important examples to reference in design as they couch the cognitive, environmental, and social elements of experience as inextricably tied together. For the HCI community, perhaps the most relevant classical phenomenologist is Maurice Merleau-Ponty, who incorporated a naturalistic understanding of the human body into their analysis of experience [453].

[108] provided a modern approach to phenomenology and technology, interpreting embodiment and phenomenology as critical to interactive design practices. Embodiment is not physical awareness, and phenomenology is more than first-person reporting—both are deeply involved with analyzing the totality of being. Dourish's work forms a foundation for including phenomenological perspectives in digital design, establishing guidance to develop a practice oriented toward embodiment and meaningful actions. [112] uses a Merleau-Pontian perspective to describe skill acquisition, pointing out that we do not acquire phenomenal information and then process it for available options, but rather that an agent "...immediately sees things from some perspective and sees them as affording a certain action".

Recently, investigations into how games are played as epistemic and ontological

activities have yielded insights into video games as culturally and historically situated frameworks for play. In Miguel Sicart's work on *Playthings*, for example, the activity of play within the framework of things represents a materially situated experience moderated by the presented affordances of the system and the disposition of the user who engages with it. This reflective nature of video game play, changed by how the user engages with the video game, represents an acknowledgment that video game play does not occur in isolation from the world (or body-mind) but rather a situated engagement between the plaything and player themselves [442].

The private industry has successfully incorporated phenomenology into product development. Good design is now popularly argued to stem from a detailed and diverse understanding of possible user experiences. Dan Norman's bible on Human-Centered Design is a hallmark of this pivot: we design for people, not for requirements [344]. Norman points out that objects with observed affordances (properties of things that indicate how they will be used) drive "good" design. The spread of human-centered design into communities, such as human-computer interaction, has produced diverse approaches to incorporating phenomenology into practical product development practices.

For example, "Persona" analysis has been a tool in software development to capture a diversity of viewpoints when designing products. This type of analysis requires a research period in which the practitioner must create a profile that reflects a semantically meaningful user category. Personas then become design and evaluation tools—how will this platonic persona use the designed system? This approach generally involves a range of interviews, reading firsthand accounts, and collecting important contextual

information about a critical customer segment to create a summarized representation of their goals and capabilities [415].

However, phenomenology in game design extends beyond persona analysis. Phenomenology was used masterfully by Liel Leibovitz to describe the parallels between deep video game play and religious exploration, discussing how our continuous conversation with the rules and mechanics of video games mirrors our larger struggles with spirituality and religion [267]. From yet another lens, David Sudnow described a descent into abstraction through engaging with the game "Breakout," providing a stage to demonstrate the depth of experience one can gain from games we might now consider simple or outdated [464]. This unique phenomenal experience is not just limited to gameplay: Game designers and engineers engage in substantially different groups of phenomenological experiences, as noted by Gladden [154]. In general, phenomenological game design practices involve a conversation between a designer and an audience surrounding a subset of qualia, informed by the usage (or imagined usage) of the designed artifact by an individual and the predispositions of the individual reporting on it. This reporting can be quantified and used in further software development, automating phenomenologically-defined quantitative profiles in advertising targeting, recommending systems, etc.

3.3 Wait, Qualia?!

In the last section I mentioned a term that may be a bit unfamiliar: *qualia*. I use this term because it reflects that the tension around understanding aspects of

a person's experiential phenomena has a rich foundation in philosophy and cognitive sciences. That is, the term "qualia" (singular—"quale") has been used to describe the non-representational "phenomenal character" of experience [492]. Qualia can be considered as the experience of sense data and mind and can only be concretely accessed through introspection of the agent who has experienced a Quale. I bring this up in the context of phenomenology (and thus, balance) because it helps capture the idea that it is not just numbers in a spreadsheet that tells us if a play experience is balanced. It is the summation of available elements of play experience tied to fairness that does so, including the emotional and sensory information tied into it. Tuning numbers is a core part of the process, but knowing how those numbers tie to the variety of phenomenological traits of a player's quale determines what balancing strategy should be used.

Laboring the point, a thought experiment consistently used in cognitive sciences that illustrates what Qualia discusses is the experience of a "color scientist" named Mary. Mary is an esteemed researcher of the color red and has spent their entire life understanding every physicalist property of color. However, Mary has never been exposed to the color red. Instead, Mary has stayed in their study, religiously pouring over black-and-white textbooks and diagrams illustrating the wave nature of light, indexes of what color objects are, and so on. One day, Mary steps outside their office and sees a red object (a rose) for the first time in their life. If Mary had perfect knowledge of the color red before seeing the rose, has Mary experienced new phenomena by experiencing the sight, smell, and texture of a red object? Most would argue that the experiential phenomena of redness fundamentally differ from the literal knowledge learned from

Mary's research [215]. While this thought problem is meant to illustrate what qualia are and are under continuous debate, it highlights how reductionist enterprises strip away seemingly critical components of experience to the researcher Mary. Many would intuitively say that Mary has experienced a new color-related phenomenon by having a new quale about redness. In this regard, we gain a new theoretical tool for investigating the phenomenon of redness: one that is non-representational, crosses and unifies sense data and is seemingly critical to the agent's being in the world.

To illustrate how this relates to balance: Brie Chin-Deyerle, a designer on the *Halo* series, mentioned how a heatmap of a specific level showed an overwhelming bias towards a single "pit" in the level for combat. The players would descend into the pit and inevitably die there, causing the area to glow bright red. On analysis of playtesting recordings, the designers noticed something interesting—it wasn't the geometry of the level causing the bias, but that players seemingly did not know how to navigate out of the pit. The pit was dark and lacked lighting cues for exit ramps, so players didn't realize they existed. A tweak of the lighting in the pit caused the combat heatmap of the level to balance out. In other words, the sensory information balanced the game environment, not the tuning of level geometry, sniper sight-lines, and so on. Returning to our understanding of balance as deeply connected to fairness, we might consider fairness as an element of qualia—it ranges from a bottoms-up feeling or top-down thought that tells someone that something is not quite right for them in the present moment.

3.4 A Framework for Designing Around Qualia

Throughout this work, you will notice that I tend to put processes in terms of theoretical frameworks and conceptual architecture. This is intentional—the specificity of audience, context, and design goals means that oftentimes getting heuristics for how we will design is of more utility than overly-opinionated implementations. The range of balancing definitions alongside the scope of potential desired player qualia means we will likely have to make bespoke solutions for every new game or audience we design for. That being said, having framing for how to approach building those bespoke solutions means we walk in with a strategy to digest a problem instead of struggling with a truly blank canvas. The frameworks I present are also always iterative to keep the "looping" nature of releasing and playtesting games. In reflection of how real-world game design works, we rarely hit the bullseye on the first dart. Instead, we throw the dart, see how it landed, adjust our aim, and try again.

As such, when we consider the multi-modal nature of qualia in game and balance design, I have put together a framework for how we can go to a system from an imagined and researched player experience. This framework consists of four major steps, each revolving around how we investigate and implement game experiences guided by the framing of design around qualia. A visual of this framework is shown in 3.1.

In the "Qualia Generation" step, the aim is to prompt the qualia we seek in play, either through engagement with the play artifact itself or through other modes of approximation. You might play *Journey*[476], or you might go into the desert to experience what it is like to frolic in the sand to inform a core game element of *Jour-*

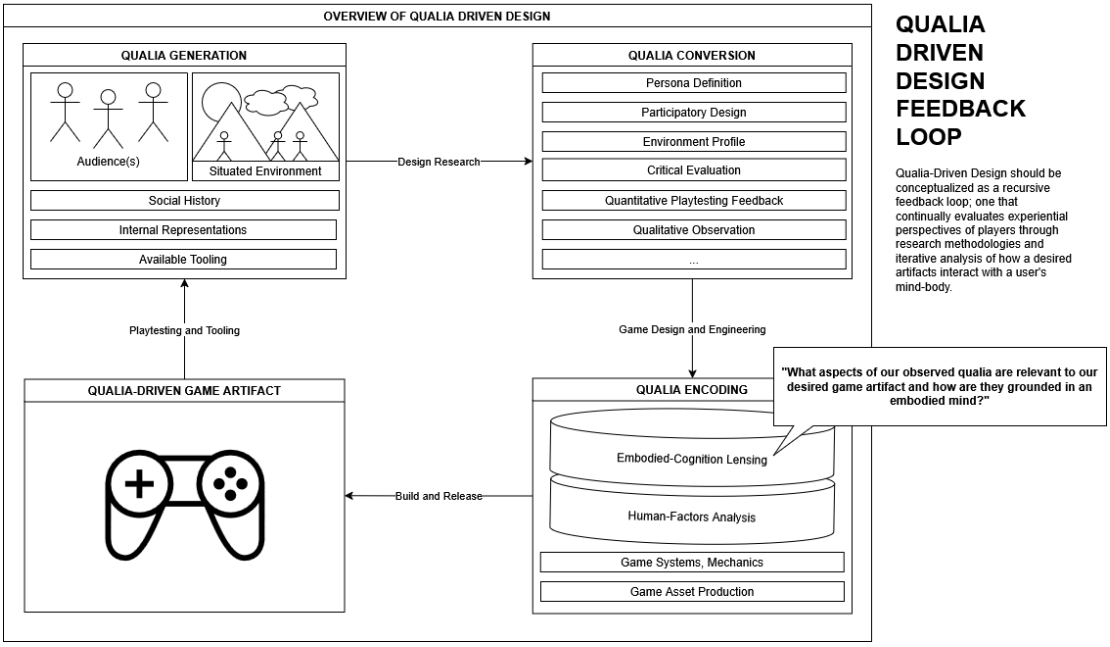

Figure 3.1: A conceptual overview of how qualia-driven design practice. It focuses on the movement from conceptualizing the context of a desired experience, performing a variety of qualitative/quantitave observations about the experience, encoding those observations in game properties, and releasing the game for playtesting. The playtesting, in turn, creates more contextual experiences to be digested in future iterations.

ney (See Figure 3.2) [116]. Aside from surface level observations, diving into qualia requires understanding the situatedness of the experience. It is not just the things seen, but the thoughts evoked and experienced, the emotions felt, the social and economic factors at play, and so on. Understanding all of this context leads to the second step of the process "Qualia Conversion". This involves using evaluation strategies to get at the kernel of those experiences—defining personas, adding in participatory design, observing playtesting. The goal is to get actionable communication about the qualia into a design language. That design language, in turn, needs to be implemented in the computational environment of the game itself—"Qualia Encoding". This step requires us to ask questions around how rule sets and play environments get at the qualia we desire to evoke. This ultimately results in an artifact that can be used iteratively to generate more rounds of qualia-driven analysis.

This framework should not be taken as a set of explicit steps that must be followed. Indeed, many designers already have the qualia from prior experience that inform their initial designs. Sometimes, this loop is very explicitly formalized, especially as it relates to playtesting-design cycles. The loop isn't meant to be taken as a novel method of working, but rather a description of how we already work, with guidance on steps we may miss out on during the design process. For the purposes of balance, especially in academic literature, it frequently is the case that researchers skip or under-emphasize the importance of "who" and "what experience" are, which is why phenomenology and qualia are so heavily emphasized in this chapter. Given the connection between balance and fairness, we can now frame and investigate fairness as an element of qualia that we are designing for when we discuss balance.

Figure 3.2: In a GDC 2013 talk, John Edwards describes how team outings to beaches and dunes allowed the translation of the entire experience into the setting of the game *Journey*.

3.5 Balance as Fairness, Fairness as Qualia

If we take that balance is deeply related to fairness, we must ask the question—what is fairness? I argue that fairness can be seen as a Quale [], something we feel and think about in our inner lives in response to some event. I am not going to claim expertise in cognitive or social science—instead, I will draw from these sources to build a holistic image of considerations we should have when thinking about fairness (and thus, balance design).

First, we can look to dual process theory from the cognitive sciences. The dual process theory proposes that humans have two distinct channels of reasoning (one fast and emotional, the other slow and logical) [34]. Although this is often generalized, this theory has a specific application in moral psychology that is relevant to our discussion of fairness. The moral variant of this theory extends fast/slow processing to the realm of moral reasoning [163]. The fast system relies on ingrained, automatic-emotional reactions to moral situations. In regards to balance, this might be felt as that feeling in your gut that "whoa, what the hell, that should not have killed me!". This intuitive response is a mixture of nature and nurture, and can be observed in young children and even animals when faced with unfair situations [38, 51]. On the other side of things, the top-down, logical side of fairness judgments come from what we know and think through in regards to a situation. In balance, this would be reflected by reviewing a tournament analytics sheet and realizing "weird, one strategy was selected 75% of the time. Something about it must be broken!" Figure 3.3 shows a simplified version of how the interaction between a game and a player "gut" and "brain" leads to the experience

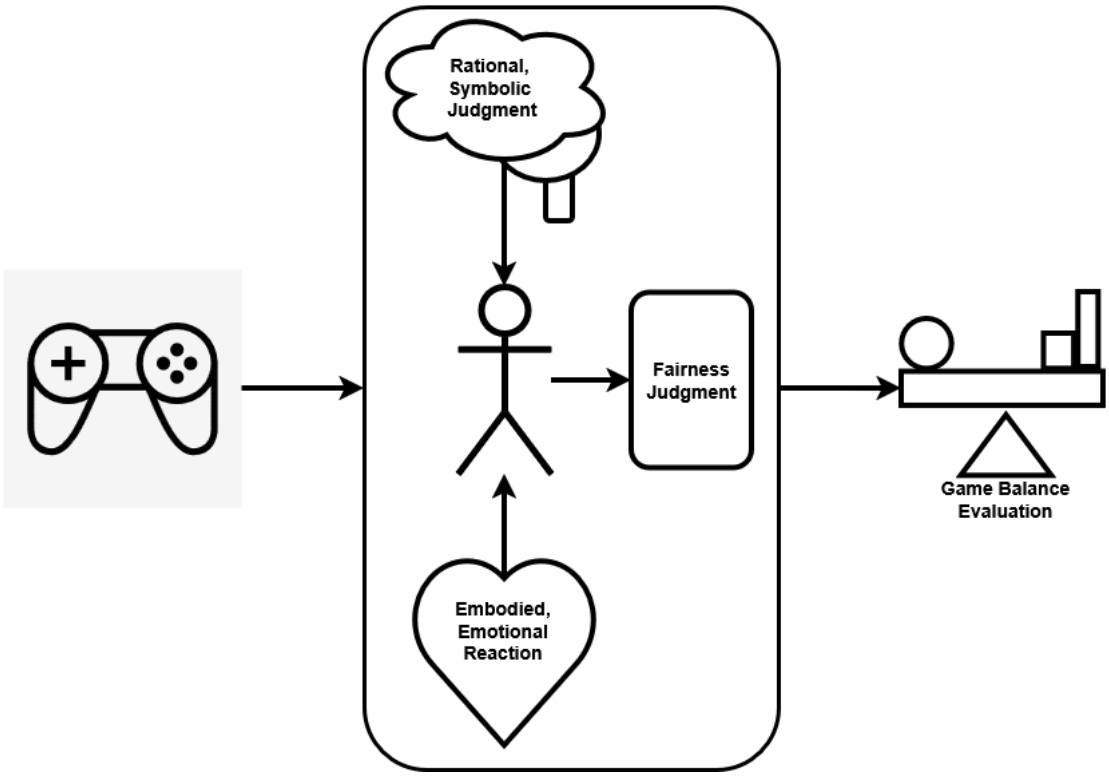

Figure 3.3: A visual demonstration of how the dual process of moral judgments is connected to game balancing. Balancing occurs from the interaction of the player and the game, with a mediated judgment of fairness of the game experience producing a balance judgment. This makes game balance and the act of a designer balancing a distinct sub-category of global game design, which may emphasize or isolate other aspects of qualia while developing a game.

of game balance.

Although early formulations of dual process theory presented these modes more as a competitive dichotomy, it evolved into an understanding as a spectrum of influence between systems with interaction between the two systems that takes place and is regulated through neurological systems [164]. The conceptualization of competing and coordinating bottoms-up moral reactions and top-down logical reasoning allows us to consider the situatedness of critical social, economic, and identity in our understandings of fairness. Our histories with our environments are encoded into our emotional

reactions, and symbolic knowledge about them can influence our moral decision making. Thus, when considering fairness/balance, our rationale for the qualia should extend beyond the immediate game environment. What does the audience of the player view as fair? What does their background tell us about what a balanced experience would be for them? Additionally, if there is an interaction between these cognitive systems, we might also consider how long-term play changes perceptions of fairness over time. An example of this might be "tilt", in which an initial failure by a player cascades into an increasingly negative affect, which causes more mistakes and reactions of frustration [95]. On the other end, a player entering a competitive scene might feel shut down by the skill levels of their opponents, but learn that a mindset that focuses on learning from mistakes instead of reacting negatively provides a far more rewarding experience for them.

It should not be taken that the dual process theory is a proven cognitive fact from my usage of it here. Instead, I view it as a helpful model for the types of visceral balance felt in someone's embodied phenomenological experience. It gives us a broad view of what factors to consider when designing for fairness and balance, which is perhaps the most important perspective to utilize when starting to design for balance. Through this lens, the seemingly white light of defining and designing for balance becomes a kaleidoscope of color and definitions, each reflecting ways in which we can frame our tuning practices. Using this lens gives us clear boundaries between game balance implementations and the larger scope of game design at large—balance considers game design as it is applied to the experience of fairness specifically.

Part II

How Do We *Design* and *Proceduralize* Balance for a Game?

Chapter 4

Taxonomizing Balance

"You're not trying to please everybody.

Balance? For who?"

Oscar Gonzales, Founder of Cool as

Heck Games

It is finally time—after gesturing broadly at what balance is, we are going to dive into getting into the nitty-gritty of what it is. As is the answer to many questions about complex topics, the answer is "it depends" at a high level. That being said, there is a clear scope we can pull out of the discussions we can observe about the topic. We can take this unwieldy bundling of concepts and contexts and (to borrow the term from Schell [420]) lens it into an organized taxonomy of terms. We need this taxonomy to capture the full landscape of how we talk about game balancing. The term loses its significance (and power, reproducibility) if we take it at face value and intuitive definitions. Using the term in this way often suggests a broadness, when in reality a narrow

subsection of balance is being explored—a taxonomy helps us build a communicative substrate through which we can make better "apples-to-apples" comparisons instead of having two ships pass in the night during design discussions.

I want to be clear in this chapter—I do not believe any singular (or multidimensional) definition of balance is wrong; I tend to think they are nearly entirely valid. Instead, I want to search for and propose a definition hierarchy that captures all of these definitions while providing a singular thread for us to talk about and focus the practice of designing (and proceduralizing) game balance. Knowing this thread not only helps us research and design in the early stages, but also gives us a footing for evaluation and development of algorithmic fitness functions better suited to our design goals.

4.1 Background

4.1.1 Balance Fragmentation

While chapter 2 gave an introduction to the diversity of balance definitions, it is worth it to get a bit more nuance on this spectrum of interpretations. To give some quick examples of the fragmentation of balance definitions: Publications will often use different definitions in different game contexts (e.g., number of players or game genre) with different evaluation criteria and methods. For instance, Olesen et al. [348] focuses on player challenge balancing in an RTS game, while Gerling et al. [150] examines balancing games to include those of varying degrees of physical ability. The diversity of definitions is exacerbated by the context in which they are used—sometimes in single-player, sometimes in multiplayer, sometimes in massively multiplayer online role-playing

games (MMORPGs), sometimes in fighting games, and so on. The work analyzed in this chapter covered a wide range of studies focused on a wide range of genres and games. For a more detailed dive into this side of the data, see appendix C. Given the diversity of definitions and contexts, game balance can be viewed as a multidimensional concept that has a separation between the various ways it is applied to different systems [422].

Even where authors acknowledge that the term is multidimensional, authors tend to give different sets of components to the term. For example, Becker and Görlich [33] breaks balance down into roughly four aspects: "...characteristics of well-balanced games, meaningful decision options, player skill, and the prevention of dominant strategies". Their analysis included 14 publications across academic and practitioner sources and used semantic analysis, noting that, although there were overlapping concepts, sources generally had different understandings of the term. Meanwhile, Pfau and Seif El-Nasr [369] settled on four categories in the context of the MMORPG *Guild Wars 2* [21]—viability, symmetry, fairness, and difficulty. Another paper considers balance from various perspectives on tuning to include players of diverse skill levels, stating "the trick is to stay behind", seemingly contradicting work that focuses on more competition- or difficulty-oriented visions of balance [157]. Schell [420] notes 13 different dualities in balance, and Fullerton et al. [138] notes 4. In developing a taxonomy, I want to capture all of these sub-definitions and ensure future nuance can be captured.

4.1.2 Values of Taxonomies in Game Design

Taxonomies are crucial to advancing science and knowledge as a whole—they provide structured terms and definitions to organize a given topic. Creating taxonomies

helps to make specialized knowledge with a level of complexity or textual diversity clear and usable to readers [259]. Taxonomies specifically help us organize and classify information in such a way that communication and collaboration between researchers and designers occur with less friction [538]. Taxonomies also enable the challenges caused by vague definitions to be addressed, while opening the door for cross-discipline interactions in research to occur [493].

Video games have had many taxonomies developed around them, aiding research in serious games [288], visual styles [81], game artificial intelligence [167], and player feedback [487], to name a few. In order for a taxonomy to be useful, it must employ rigorous methods, including techniques such as literature reviews and coding exercises to ensure that they accurately reflect a specific area of research [345, 336]. As a taxonomy for game balance employing these techniques does not yet exist, there is a marked opportunity to improve how research is performed in the domain of video game balancing.

4.1.3 Game Balance vs. Game Balancing Strategies

There is something important that I want to preface before continuing this chapter—delineating definition and practice. There is a marked difference between the strategies used to balance games and the definitions of balance that are focused on. A prominent example of this is Dynamic Difficulty Adjustment (DDA), which aims to adjust difficulty for a player based on their current level of engagement. While some might be tempted to define this as a category in and of itself, it can be applied to a wide variety of differing use cases: adjustment for personality traits [481, 322, 182], diffi-

culty [126, 198, 182, 447], economies [126, 198], or player inclusion [26]. Automatic Game Balancing (AGB) has similar traits and leverages many sub-approaches—reinforcement learning to balance for difficulty [14] or player inclusion [387], deep player modeling for variability of player choice and strategy [370], or evolutionary algorithms to achieve variability of strategy while maintaining a balanced competitive landscape [323, 433]. The content of this work does not attempt to taxonomize specific methods and targets of balancing as they can be used in a wide variety of balancing definitions, instead focusing on the high-level goals and definitions of balance as a whole. An important distinction to note in the game balancing strategies is between static balancing and dynamic balancing. Static balancing refers to strategies that are used during the design of a system but do not change in response to player actions. Dynamic balancing refers to the runtime alteration of a game to impact the performance of a player [16].

While I put aside balancing strategies explicitly in this chapter, the next two chapters focus on the subject—stay tuned.

4.2 Methodology

As highlighted in previous sections, there is a wide variety of balance research in academia. Any given set of papers will operationalize balance differently, use a myriad of techniques to achieve it, or evaluate various related outcomes. In order to encapsulate this diverse range of perspectives and identify the varied aspects of game balance, we (myself, my advisor, and two undergraduate students) constructed a taxonomy of game balance. First, we conducted a comprehensive literature review of 95 academic papers

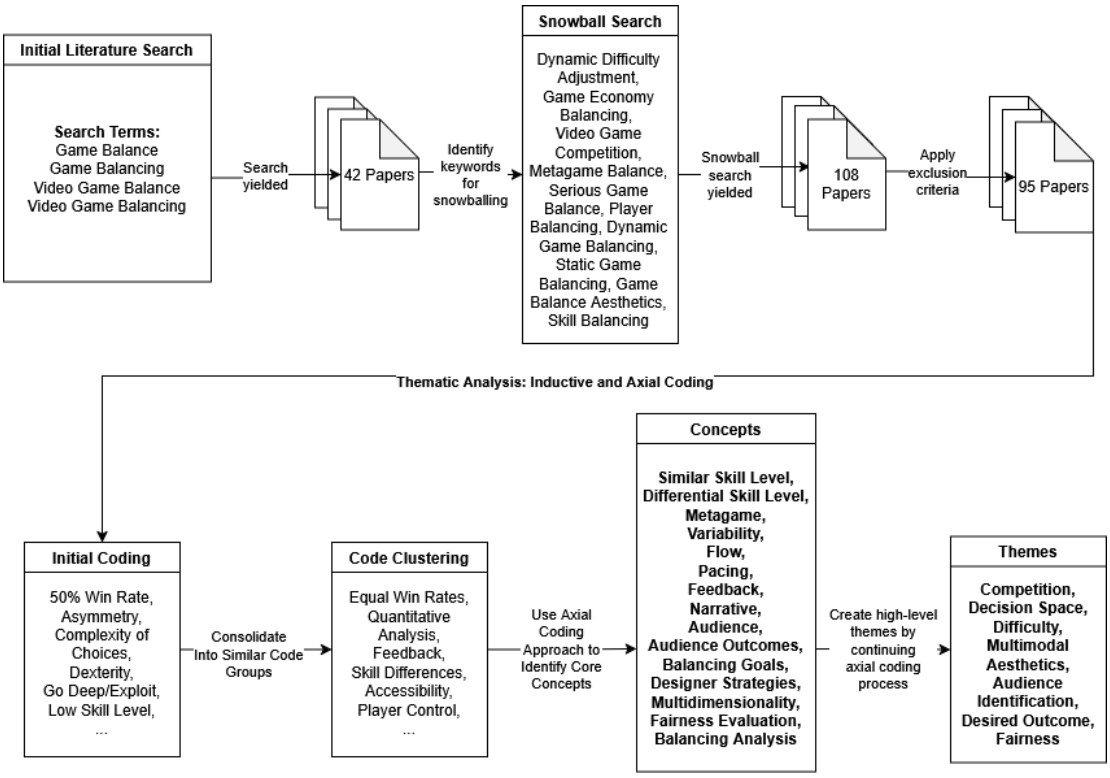

Figure 4.1: An overview of our literature collection and thematic analysis process.

that define balance. We then performed an inductive thematic analysis [413] to identify and group related codes and concepts that arose from the corpus. This resulted in four key themes with eight underlying concepts. We detail the methodology and process in more detail below.

4.2.1 Methodology

Search Strategy and Data Collection

For the search strategy, we identified an initial set of relevant terms for game balance to search within an academic database. The initial list of search terms we identified were: "Game Balancing", "Game Balance", "Video Game Balancing", and

"Video Game Balance". Once the search terms were finalized, we searched Google Scholar for each term individually and reviewed the first 100 results. Importantly, we restricted the dates of the search from January 2000 to December 2024, as this is the time frame when most literature on game balance emerged. Any publication that appeared relevant through title, abstract, or key words was collected and added to our game balance corpus. This process yielded an initial set of 42 publications.

After our initial pass, we then utilized snowball sampling by reviewing the collected publications and identifying any relevant key terms that could also be used to perform iterative searches [328]. Examples of terms used for subsequent searches were "Dynamic Difficulty Adjustment", "Game Skill Balancing", "Game Economy Balancing", and "Game Balance Definition". See figure 4.1 for the full list of search terms utilized. Each of these subsequent searches also reviewed the first 100 results, where publications unrelated to game balance or already collected were excluded. At the end of all searching, we ended up with 108 total publications in our corpus for review.

We then validated and refined the corpus by identifying exclusion criteria and performing a final pass to remove any papers that did not meet the criteria. Namely, our criteria excluded any work that 1) referred to physical balance instead of game balance, 2) was not peer-reviewed (e.g. Dissertations or Master's Theses) or unrelated to game development, and 3) was a duplicate of another work already in the corpus. This resulted in a final list of 95 publications for analysis.

Inductive Thematic Analysis

In order to analyze the corpus and generate a taxonomy of game balance, our team of researchers performed an inductive thematic analysis [84, 329] on the definitions and evaluation methods for game balance provided in each paper. We chose to use inductive thematic analysis because its bottom-up approach is best suited to capture the wide range of theories and reasoning underlying the many different definitions of game balance. Inductive thematic analysis is also particularly useful when exploring new terrain [84], such as a taxonomic understanding of balance. We followed an inductive thematic analysis approach by performing the following steps: familiarizing with data, generating initial codes, iteratively grouping codes and searching for themes/concepts, and reviewing and defining the themes [329]. A visual representation of our overall methodology and process can be seen in figure 4.1. We discuss each step in more detail below:

- **Familiarization with Data**—Three members of the research team went through each paper in the corpus to extract relevant text on game balance. They specifically targeted the introduction, related work, and discussion sections of publications, as they most frequently included explicit definitions of balance along with an identification of the important considerations and impacts of balancing a game. The team paid special attention to the evaluation sections of the publications as they demonstrate which key concepts are relevant to balance based on how they measure success quantitatively or qualitatively. The team also reviewed several game designer videos on Youtube (*Game Maker's Toolkit* [488], *Masahiro Sakurai*

on Creating Games [349], *Adam Millard—The Architect of Games* [347]) to give a strong primer on game balancing.

- **Generation of Initial Codes**—The research team collectively performed an inductive coding process, reviewing the collected publications for key terms related to game balancing (e.g. "challenge", "low-skilled", "fairness"). Each publication would produce a list of unique codes. Repeated usage of the same term within a paper was only counted once for that paper. This coding process resulted in a raw count of 1952 terms across all papers. These terms were then counted and consolidated into a list of 942 unique terms. Consolidation was performed using a simple Python script that counted repeated terms and created a frequency count for terms with the exact same wording and spelling. A link to this script can be found in the appendix. The team found that terms were classified into one of three categories: definition (relating to defining or describing the balancing process and its outcomes), method (related to the purely procedural approach used to achieve some aspect of balance), and target (related to a specific game element such as aiming that could be employed during a balance adjustment). For the purposes of this work, we focused on codes related to definitions, as methods and targets were definition agnostic and did not help inform what subcategories of balance might exist. By eliminating terms that exclusively related to methods and targets, we were left with 527 unique codes that had 1196 total appearances in the collected publications.

- **Iterative Code Grouping and Searching for Themes / Concepts**—From

the finalized list of initial codes, a round of clustering via axial coding [424] was performed to consolidate similar or related codes into a smaller list of unique codes. For example, the codes "close matches", "successful scoring events are close", "narrow score margins", "similar scores", and "similar success rate from all players" were combined into the new code "close-scoring matches". All instances of each constituent code were summed, providing an appearance frequency for each new clustered code (e.g., "skill" codes ended up with 43 total appearances after summing similar terms). This produced a list of 105 "clustered" codes. We then repeated this clustering process until the codes were refined enough to identify key game balance concepts. The end result of this phase identified 15 key concepts that formed seven overarching themes.

- **Reviewing and Defining Themes**—When reviewing concepts and themes, the team identified that they focused on two uniquely different aspects: taxonomic and contextual. Taxonomic refers to explicit descriptions of game balance and how it is defined. Contextual refers to the context in which balance applies (i.e., its audience identification, desired outcomes, and fairness). The taxonomic category contained four themes and eight underlying concepts with a total of 65% of code appearances, while the context category contained three themes and seven concepts with a total of 35% of code appearances. For the purposes of this work, we chose to only include the taxonomic aspects due to the wide generalization of contextual codes, which could be applied to any given definition of game balance.

A Sankey diagram showing the consolidation of terms can be seen in figure

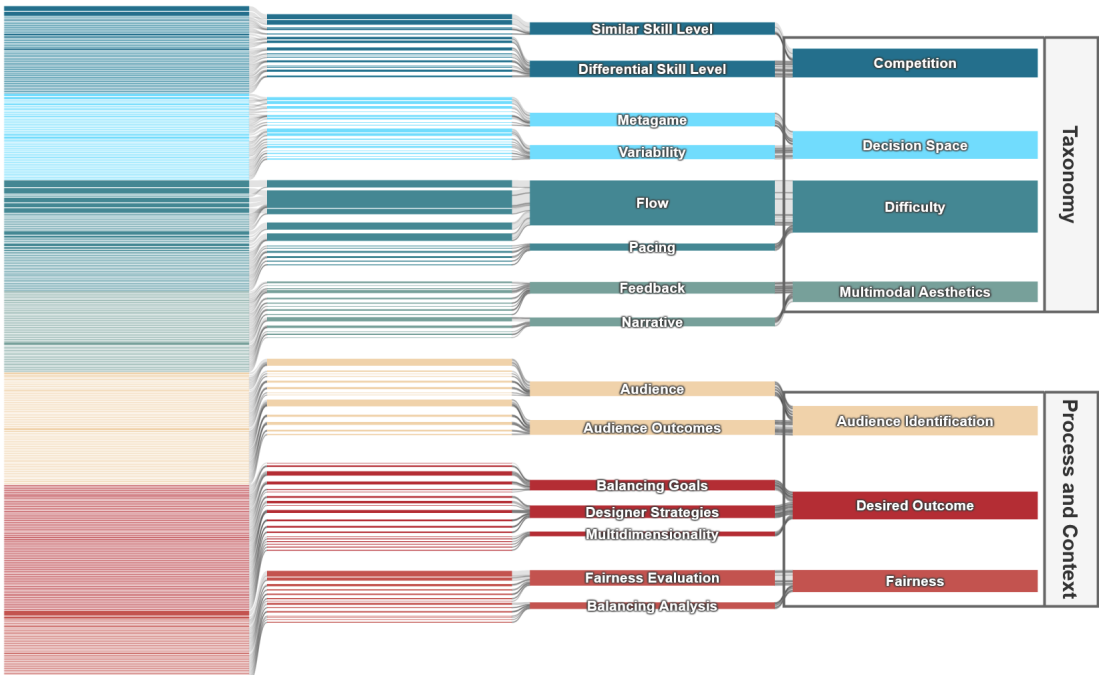

Figure 4.2: A Sankey diagram showing the consolidation from codes to code groupings to concepts to themes. There are 15 total concepts and seven final themes. The bottom three themes relate to the general process and context of game balance, while the top four themes constitute key taxonomic categories of game balance definition.

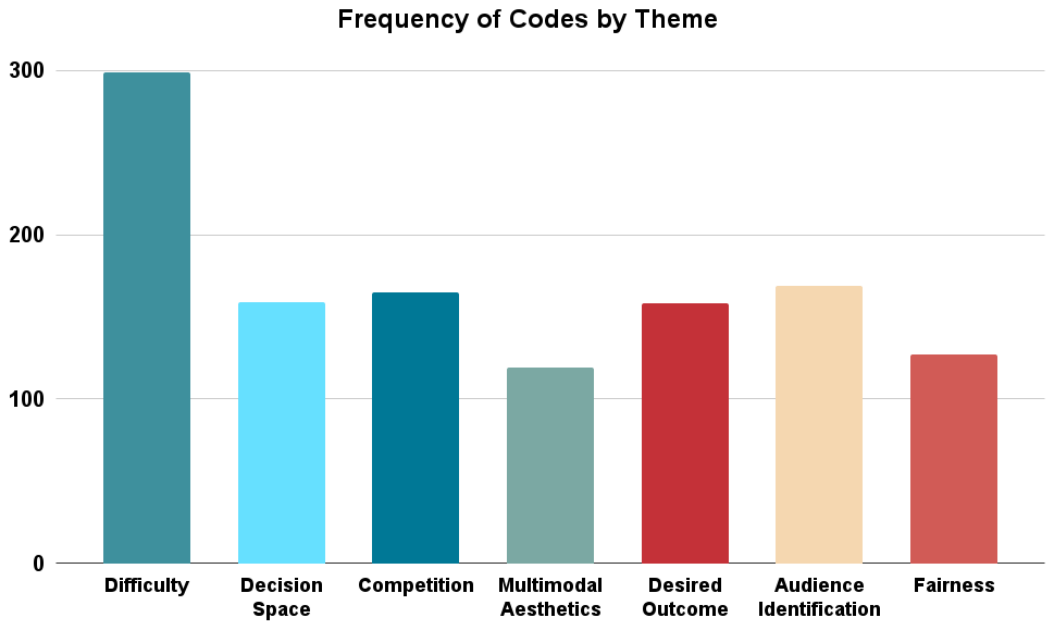

Figure 4.3: A bar graph showing the frequency of codes sorted by theme. Difficulty has the highest allocation by a large margin.

4.2, and a bar graph showing the frequency of codes against each group can be seen in figure 4.3. A link to the full dataset and tooling used to visualize the data can be found in the appendix.

Positionality Statement

The literature search was conducted by four members of the research team: two undergraduates (one male, one female), one PhD candidate (male), and one professor (male). All members of the research team came from the same institution and department, which is focused on media development and games. All four members had prior experience playing and developing games, while the PhD candidate and professor had prior HCI and game research experience. The thematic analysis was performed by

three members of the original four person team—one undergraduate (female), one PhD candidate (male), and one professor (male).

4.3 Results—Taxonomy of Game Balance

Our taxonomy indicates that there are four key themes underlying the definitions of game balance: **competition**, **difficulty**, **multimodal aesthetics**, and the **decision space**. These represent the multidimensionality of balancing, showing that there are multiple possible ways to define balance (and subsequently implement and evaluate it) depending on the context of the game and desired goals. These taxonomic categories are further broken down into eight underlying concepts, providing a more nuanced understanding of the key aspects of each theme and the breadth of design focus that game balance addresses overall. In the following subsections, we review each category/theme of the game balance taxonomy as well as their underlying concepts. A visual representation of our taxonomy can be seen in figure 4.4.

As mentioned in the previous section, we also identified four additional themes related to game balancing: **desired outcome**, **audience identification**, and **fairness**. However, these categories are more focused on context and informing the design of game balance as a whole, rather than explicitly differentiating the various aspects of game balance. As a result, they are more appropriate as general considerations for the design of balance than as taxonomic categories themselves. See section 4.3.1 for a more in-depth discussion of these context themes that we identified but ultimately did not include in the taxonomy presented below.

Through our results, I propose a central definition for balance that acts as the root for all sub-definitions presented in the coming results:

Game balance is the result of *designing game systems* to have a *level of perceived fairness* for a *given audience* based on a *designer's goals*.

This forms the core of all other specific definitions of balance, with careful wording to cover edge cases and conflicts I have noted up until this point:

1. *"...designing game systems..."*: This signals that balancing is a design process when building a game to get the result of a balanced game. Designing in this manner encapsulates modification of parameters, tuning, and so on.

2. *"...level of perceived fairness..."*: Balance is oriented around the qualia of fairness—but the goal of balance is not to make every experience feel positively fair. Asymmetric balance, hidden balancing, punishing games all point to unfair situations still considered as game balance.

3. *"...given audience..."*: Fairness is subjective and internal; one individual (or set of similar individuals) will have wildly different understandings of what is fair. Saying that the sense of fairness in a lifelong gamer is the same as someone who has never picked up a controller is foolhardy. Whether implicit or explicit, the identities of our players have an impact on how balance is understood.

4. *"...designer goals..."*: Balance (and fairness) as an emergent outcome means that the designer is trying to say something through their balance design. A party game might seek to playfully include, a horror game might seek to make things

feel insurmountable or out of control. Determining what sense of fairness is being designed for aesthetically is crucial for every downstream approach to balance.

4.3.1 The Core of Balance—Desired Outcome, Audience Identification, and Fairness

Through the methodology presented, we ended up with seven core themes. While four of these refer to sub-definitions of balance (Difficulty, Decision Space, Competition, Multimodal Aesthetics), the other three (Desired Outcome, Audience Identification, Fairness) ended up applying to game balance regardless of which definition of balance you apply. From this, I reasoned that their universality points to the foundation of game balance and balancing practices—designers want some desired outcome for an audience that revolves around fairness. This gives us the generic definition we covered above. Through specific applications of this central principle, we get the diverse range of balancing practices we will detail below and in the following chapters.

Fairness

This balance context draws on the common idea throughout most of the literature that suggests a prevalent thread of balancing systems is creating a sense of **fairness** in users while they play. Fairness is subjective and just as hard to pin down as balancing and, as such, requires consideration of the goals, audience, and taxonomic direction of their system in order to evaluate clearly. This is not only important to understand balancing success in an audience, but also crucial to understand while implementing automated or generative approaches, as many of these approaches require converting

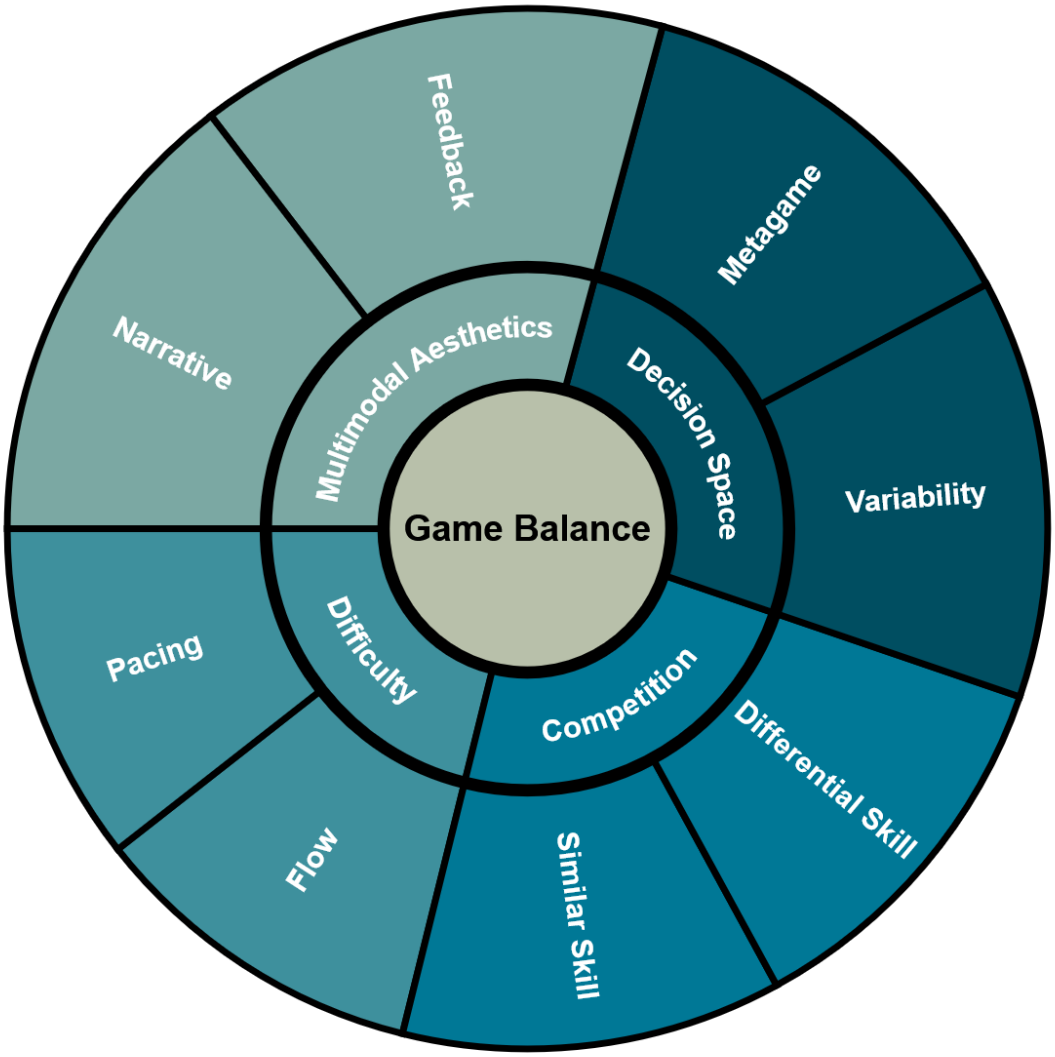

Figure 4.4: A circular graphing of a taxonomy of game balance. The inner circle has the four main taxonomic categories—**difficulty**, **multimodal aesthetics**, **decision space**, and **competition**. The outer circle has the eight subcategories—*narrative*, *feedback*, *metagame*, *variability*, *differential skill level*, *similar skill level*, *flow*, and *pacing*. The center of the ring represents the core definition of balance elaborated in 4.3.1

what the designer thinks is an important evaluation metric into a fitness function that can be used for automated scoring of different system outcomes; especially in genetic and neuroevolution techniques [268, 13, 104, 378, 391].

Audience Identification

This balance context is critical to recognize, as the preferences of the user will directly impact how the balancing system is perceived [481, 30, 186]. Even in terms of straightforward concepts like *flow*, audience identification can have major impacts on balancing outcomes. For example, Baumann et al. [30] posits that "...there are considerable individuals differences in the extent to which people seek and are able to self-regulate experiences of flow and enjoyment". As such, someone designing a game like *Mario Party* [196] might want to target balance for audiences with a wide variety of skill levels, as families of different ages or large and diverse friend groups would be intended to play together. This would indicate that they are balancing in the *differential skill level* concept of the **competition** theme and need to adjust their design considerations accordingly. As such, audience feedback from a wide range of skill levels playing together should be of higher priority than those who prefer games where increased mastery and experience give them distinct advantages over other players. Even beyond balancing within the game system itself, audience understanding is critical to modulating how balanced a game is as perceived by its players. Jeff Kaplan stated in a blog post concerning the balance of *Overwatch* [41] that "the perception of balance is more powerful than balance itself" [231].

Desired Outcome

This balance context relates to the specific measure of what a designer intends to accomplish in the context of a specific game as it relates to balance [506, 36, 378, 235, 70]. This encompasses the themes of:

1. *Balancing goals*— What are the intended outcomes of balancing?
2. *Multidimensionality*—What are the relevant categorical themes for the balancing goals?
3. *Designer Strategies*—What is the appropriate design method to achieve balancing goals in context of relevant themes?

Using these themes is critical to understand what outcomes we want from the game balancing process and applies to all contexts of game balance. As Beyer et al. [36] states—"Game balancing is the process of finding a rule-set and corresponding parameters such that the resulting gameplay defined accordingly satisfies certain design goals, such as fairness." Setting balancing goals in particular helps understand why a balancing system is being implemented in the first place—are you looking for a positive affect in your users, developing a deep and rewarding competitive eSports scene, or ensuring that one balancing system interacts gracefully with another? Answering these questions helps set the tone for the system and helps identify which areas of the balance taxonomy to focus on during design and development.

4.3.2 Difficulty

The category of **difficulty** in game balance refers to setting the challenge of the game to appropriately address a player's capabilities [500]. For instance, Andrade et al. [16] puts it this way: "Balancing a game consists in changing parameters, scenarios and behaviors in order to avoid the extremes of getting the player frustrated because the game is too hard or becoming bored because the game is too easy." As such, when considering game balance in the scope of difficulty, careful consideration should be placed on establishing the long-term *pacing* of the game and short-term management of the player's sense of *flow*.

Pacing

Pacing is the long-term establishment of player progression throughout gameplay and is related to the sense and speed of progress [30]. This is a key aspect underlying difficulty as games can directly control the speed of pacing to increase or decrease the sense of difficulty that a player faces over a period of time. There are a variety of approaches to control pacing, such as increasing/reducing the number of enemies or obstacles [312, 97], adjusting enemy difficulty through changes in behavior or health points [184], using a difficulty curve or difficulty spikes to modulate pace as the player gains mastery [33], and increasing the available areas to explore or abilities required to progress as the player character increases in strength and capability. Importantly, accessibility options can also affect the pace of games—Celeste [294], for example, provides a wide range of alterations in game rules that can be used to better fit the difficulty desires of the player [243].

An important area of pacing that frequently arose from the corpus is the modulation of economic game systems. These control the pace of rewards, relating to in game currency, resources, and experience [126]. Ranandeh and Mirza-Babaei [379] summarizes this concept as "...economic systems that are fun for both newcomers and experienced players, maintaining a balance between supply and demand, preventing hyperinflation, establishing effective exchange mechanisms, and number and purpose of economic resources." Rupp and Eckert [407] states that "a game's balance is thereby heavily influenced by the design and configuration of its internal economy which defines how virtual resources are created and can be transitioned to other resources."

Economic control of pacing as well as enemy volume and difficulty also provide many dynamic ways of controlling pacing throughout the game. For instance, a game might watch a player's failure rate and dynamically control the volume or placement of enemies (such as in Resident Evil 4 [67], where repeated game over screens changes enemy distribution) or the placement of resources to manage tension of players in a team (such as in Left 4 Dead [498] or the Hamlet prototype [198], where power-ups and rewards are dynamically distributed in response to player performance.

Flow

Flow relates to managing the difficulty of a game to keep a player in a flow state [331] by keeping the challenge balanced between too frustrating and too boring [30, 262, 26]. Notably, flow was the single most frequent term in the balance literature (101 occurrences), and is often the default approach for understanding how and why balance relates to challenge and difficulty across a game. Flow can be viewed as the short-term,

immediately perceivable version of overall pacing—i.e., it deals with the current affect of the player in response to obstacles being presented to them at that moment [262].

Although flow is mentioned in relation to static balancing approaches, it is a major focus of dynamic balancing, especially in terms of dynamic difficulty adjustment. As dynamic difficulty adjustment is one of the larger focuses of the overall academic literature on game balance, it is frequently used both as a design technique and as an evaluation mechanism [30] for engagement and positive impact outcomes in users, as higher levels of flow are hypothesized to result in better player experiences [472]. Dynamic approaches to flow need to consider player performance in some quantified metric, then provide adjustments to game elements, behaviors, and rule sets according to those metrics in the hopes of correctly hitting the right balance between frustration and challenge.

Achieving flow also depends greatly on the audience for which the game is designed [503]. For instance, fans of the "Soulslike" genre might appreciate a higher difficulty curve with frequent failure states that might be frustrating to other players. In contrast, family-oriented games such as the Mario franchise provide gentle learning and difficulty curves that are suitable for novice players but may be perceived as too easy for players who want a higher level of challenge.

4.3.3 Competition

Competition involves understanding game balance from the perspective of tuning a game so that the game appears fair in context of player skills in environments where players can be compared to one another. This category is important as

it addresses game environments that are intended to highly trained competitors (e.g., eSports), but also in cases where players of varying experience and ability are being compared to one another. This can exist on a spectrum, with one end prioritizing equal win-rates between players of similar skill, while the other end deals with prioritizing equal win-rates between players with gaps in skill and/or ability. This category is relatively split in the literature (nearly equal appearances of codes—94 for "Differential Skill Level" and 71 for "Similar Skill Level"). These works broadly include discussions of how to balance games so that one player does not have an unfair advantage, how to balance games to include people with varying experience, and how to design games that are inclusive to people with different abilities.

Differential Skill Level

Differential Skill Level is a concept under competition that focuses on adjusting difficulty to include players of differing skill levels such that they have the ability to reasonably compete with one another. Vicencio-Moreira et al. [503] claim "Player balancing is a solution to this problem; it involves making different adjustments to the game mechanics for different people, in order to help equalize the performance of stronger and weaker players." This definition is sometimes referred to in context of rubber-banding implementations, where player abilities are inversely moderated based on their current performance (rewards if performing poorly, punishments if performing optimally). A commonly used implementation of this is in the *Mario Kart* [114] series, where players in last place receive markedly more powerful items and speed boosts than those in first place.

Crucially, differential skill level can apply to both learned and innate ability, allowing for the support of inclusivity in game design. For example, Gerling et al. [150] explored balancing a rhythm game focused on dexterity to include players in wheelchairs. A static approach to this style of balancing is providing handicaps, mirroring sports such as golf to proactively give lower-skilled athletes a chance to compare with higher-skilled athletes [29]. Dynamic approaches to inclusion could moderate character abilities in response to performance compared to peers (such as in the rubber-banding example above) in real time. Inclusive approaches are frequently applied in contexts where the diversity of player skill is difficult to predict, yet the developers intend for all to be able to jump in and enjoy themselves with relative ease.

Similar Skill Level

Similar Skill Level is when games have difficulty tuned so that players of equal skill can compete with each other with a relatively equal win rate. Beau and Bakkes [31] provides the definition that balance "...entails that every player, given they possess equal skill, should have the same probability of winning the game." An important distinction from the differential skill level concept is that similar skill level aims to have players of equal skill have equal win rates—but a higher skilled player should frequently win against a low skilled player. This definition is particularly relevant to eSports environments, where players expect to be performing on an even playing field. Classical symmetric games with open information, such as chess and go, set a baseline for this style of balance, as it equally provides game strategies and information to players, putting player skill in the foreground. This is evidenced by [486], which uses balancing strategies

to attempt to select competitively and aesthetically balanced alternative versions of Chess. Competition involves having a deep-enough play space such that mastery and development of novel strategies are ongoing and constant, providing depth, counter-picking, and a rewarding sense of skill development over time.

4.3.4 Decision Space

The **decision space** taxonomic category refers to the ability of players to make a wide variety of meaningful decisions that do not produce large power imbalances depending on the specific strategy selected by a player. The goal of this style of balance is to provide both a wide range of options to suit different playstyles. As such, exercises in implementing and tuning games with a large decision space involve analyzing overall performance of different strategy selections against one another while also providing clear failure states for ineffective strategies so that players can learn what is a valid or optimal strategy to use in a game environment.

Variability

Variability concerns the existence of a variety of game mechanics and progression paths such that pursuing different paths involves meaningful choices. Touching on concepts from both the "similar skill level" and "differential skill level" competition subcategories, it seeks to include players who pursue different sets of experiences and play expression while maintaining the difficulty appropriate to a given game context. The management of underpowered and overpowered strategies is often emphasized in this category, as having these strategies undermine the ability of a player to pick strategies

that suit them and still be successful in the game. In a discussion on "Hearthstone" [121] deckbuilding, de Mesentier Silva et al. [101] states the danger of a lack of variability balance: "In an unbalanced game, there may be a single degenerate strategy that is unbeatable or it may be impossible to play a particular character class well. Such a game is essentially broken and experienced players tend to quickly lose interest." This variability promotes healthy competition as well as enables player expression and support for different player dispositions as they can select strategies, controls, and gameplay that matches their preferences without experiencing notable differences in challenge. Player expression, in this context, denotes how a player navigates through a range of meaningful game choices and playstyles according to their desires instead of being confined to a singular or repetitive approach [510].

Variability can be achieved through several methods, e.g., introducing a range of playable characters, items, abilities, or control strategies [217]. The ability to create permutations of these game elements increases the combinatoric possibilities of the play space. This increased combinatoric complexity in turn emphasizes player expression and making part of a given game about explicit performance. Naturally, creating a game with this level of complexity and depth can exponentially explode the number of branches in a system that need to be evaluated. This complexity is frequently addressed by proposing procedural or simulated approaches (such as [390] in the context of Pokemon [142] team building) to rapidly test a wide decision space before launching an initial version [409].

Metagame

The interactions between community members and developers can change the experience of the game. This also can happen as a result of interactions between members of a game's community, as well as from independent observations from an individual over time. All of these impacts are related to the *metagame* mode of decision space. Reis et al. [390] differentiates this from the form of balancing in run-time, providing "metagame balance is an inverse process where strategies are stimulated to be selected by players before a game starts by tuning game elements." This definition points to both the relative strength of a given strategy and the selection process by players in individual and community settings. It is important to note that while the literal and quantitative measures of a game certainly impact a metagame, this concept is more directly related to the perception of a player base and the observations collected over time.

The metagame often depends on having a constituent community playing the game, communicating with one another, and communicating with the developers of a given game. Players, through gameplay and the ensuing conversation, identify decision strategies that are most likely to find success in a game environment given their goals. This conversation can not only focus on identifying optimal strategies, but also create new modes of gameplay to fit new criteria. For example, the Magic: the Gathering [395] community has created formats such as "pauper", where rare and expensive cards are excluded from the gameplay environment to focus on a more budget-friendly yet competitive play environment. Methods for addressing metagame often rely on collecting

and analyzing gameplay metrics and tying them to player personas, specific decision space strategies, and win/loss rates. For example, Pfau and Seif El-Nasr [369] utilized deep-player analysis in the game Guild Wars 2 [21] to evaluate where weaknesses and strengths existed across class selections and their frequency of selection in a community. These weaknesses, strengths, and selection rates are then addressed through ongoing patches, "buffing" and "nerfing" different game elements to try and achieve an equilibrium [222].

4.3.5 Multimodal Aesthetics

The **multimodal aesthetics** category focuses on the aspects of balance that depend on the interplay of aesthetic qualities such as narrative beats, rule cohesiveness, enjoyability, or visual and audio feedback. Multimodal aesthetics is a somewhat broad category, but generally relates to the overall "harmony" of game elements in a wide spectrum of categories. Loebel and Koubek [287] phrases it thus: "Video games can be balanced in several aspects, such as presentation, narration, game experience and game mechanics." In essence, this dimension of balance focuses on delivering the aesthetic rewards that an audience might desire. For instance, horror games might restrict resources and feature high spikes of visual and audio feedback, while a Cozy game might give a consistent curve of rewards and audio/visual feedback. Aesthetics also underlie many gameplay mechanisms, e.g., a game economy can provide aesthetic rewards by increasing character power which in turn provides more pleasing visual and audio feedback, or a high-skill performance might be rewarded with player achievements and titles to show off to other players.

Narrative

While aesthetics can apply to a wide range of sensory and emotional modalities, an important sub-category of this definition relies on how gameplay *narrative* relates to game balance. That is, does the progression of the player and the interaction with the game systems affect the impact of the narrative of the game [198]. Related to pacing, but specifically in the dimension of storytelling, narrative rewards are given to the player based on the completion of key game requirements or triggering of game events. The aspect of balance related to this revolves around the complexity, progression, and desires of the audience to see a story develop in response to gameplay interactions.

To give an example of how narratives are balanced against gameplay, Silva et al. [446] highlights how tension is managed between hardcoded and emergent narratives and gameplay. They point out that there is a common failure in the usage of cut scenes in games, as they cut momentum created from the interactivity of the game. However, there is nuance in this balance—a game whose main draw is the narrative itself, the narrative will justify further engagement with game systems. Juul [230] focuses on this interplay between interactivity and narrative, talking about how the two will both struggle against one another and bolster one another, suggesting that balance in narrative and gameplay and careful attention to how they fit with one another is critical to the overall experience of the game. The potential for discord between mechanics and narrative is termed "ludonarrative dissonance", and is commonly discussed in both academic and industry design literature [512, 49].

Narrative aesthetic balance has many expressions throughout game develop-

ment. It can be as simple as providing story cut scenes after a segment of gameplay or as complex as simulating social relationships across a range of characters and tracking the cascading effects of interacting with characters in positive or negative light [485]. Narrative balance is deeply tied to engaging a player's sense of role-play, as providing too minimal of a narrative balance can distance the player from story elements, and too maximal of a narrative balance can cause players to be fatigued or bored by a lack of gameplay interaction.

Feedback

Aesthetics deal greatly with the *feedback* given to a player in reaction to their actions. These signal successes and failures of players and can be used as indicators to a player of how to adjust their performance so that they can better meet challenge and skill requirements of a game. This feedback is provided through multiple dimensions: game-feel elements indicating hit contact composed of visual animations and audio clips [474], playing of failure cut scenes (such as the "YOU DIED" screen in Dark Souls [135], the playing of success music (the Final Fantasy [459] battle-won music), or the increase in score or ranking of a player.

Feedback is generally statically defined in nature—it is defined as part of a core game loop prior to gameplay in response to player success. However, there are instances where feedback can also be dynamic, providing boosts to succeeding players or failing players based on the desired style of feedback loop [32]. Aesthetic feedback can be used to punctuate major successes of players for key actions—for example, in Super Smash Bros. Ultimate [28], scoring a game-winning hit that causes a knock-out

will sometimes trigger a dramatic and time-slowed zoom in of the camera, showing the hit and its corresponding particle effects in great detail to both players.

Well-tuned feedback shapes the player experience and outcomes as a whole. Feedback helps tie in systematic and mechanical changes to the overall multimodal nature of the game. This coupling of aesthetics to mechanics helps intelligibility of the game and helps the player feel engaged, rewarded, and aware of what changes in behavior they might have to employ to have a better outcome. Some authors tie successful balancing in feedback to a sense of player's self-esteem [176]. Positive affect outcomes are also tied to correctly combining aesthetic and mechanical balancing in regards to feedback—immersion [322], engagement [98], satisfaction [348], among others.

4.4 Discussion

Our results indicate that while balance is a complex and multidimensional concept, it does have a digestible set of core concepts that can be used to ground an understanding of balance, which can in turn inform design and evaluation strategy. Taxonomic definitions can help focus the balance of the game on what the game is, who is playing it, and what the designers intended the audience to experience. Our discussion focuses on helping illustrate how to read and leverage the taxonomy, explains the multi-dimensionality of balance, and touches on global aspects of balance that also fell out of the literature review.

4.4.1 Reading the Taxonomy

The taxonomy (figure 4.4) separates out four categorical themes of balance, but also suggests relationships between the categories that can help illustrate what considerations might need to be taken when examining any given concept. It should be noted that category boundaries should be viewed as somewhat porous: while there are clear topics that fall under each theme category, some may have interactions with a different thematic category.

Our taxonomy of game balance has two layers: the inner circle represents our four core theme categories, while the outer layer represents important constituent concepts that can be used to further detail a given theme of balance. Therefore, when considering how to understand a balancing system for a game, one should use this taxonomy to decide which specific dimension they are focusing on by moving from the inner circle to the outer circle of the diagram. For example, when understanding balance around different levels of skill, one might move from the **competition** theme to *the differential player skill* concept. This would emphasize the need to design and evaluate for contexts where multiple skill levels should feel welcome and competent when playing together in a game. Likewise, moving towards *similar skill levels* instead would instead focus on having success in a game directly tied to having equally skilled players trade wins equally, but a high-skilled player should have an advantage over a lower-skilled player.

In addition to defining dimensions that can be used for design or analysis of balance, the diagram is ordered so that the relative importance of a category shows the

most interaction with its immediate neighbor. For example, **competition** is adjacent to **difficulty**. This is intentional as it signals that while balancing for different contexts of skill can have different balancing approaches, the interaction with skill profiles and challenge mean that these two categories likely need to be considered in tandem based on a game's goal. Similarly, the proximity between **competition** and **decision space** links how the breadth and depth of meaningful decisions to be made by the player is in part dependent on the skill of the player to identify, select and iterate strategies in response to a complex game environment. On the other side of the circle, **competition** opposes **multimodal aesthetics**. Although no thematic category is completely independent, this placement suggests that **multimodal aesthetics** might have a lower direct impact on accounting for player skill. Aesthetics can be used to telegraph some information, but the results of evaluating a decision space and reacting to difficulty provide a much greater influence on the **competition** theme than aesthetics. On the other hand, **multimodal aesthetics** can play a greater role in **decision space** (such as providing different character skins or disclosing different levels of information to the player) [176]. Similarly, aesthetics can influence difficulty by modulating sensory rewards for the player over time [272], or by adjusting valuable signals to the player such as telegraphing attacks [463, 385].

Ultimately, using this taxonomic approach helps focus both design and research on game balance. In design, it helps break the understanding of balance problems into discrete pieces that can then be iterated upon for an ultimate desired player experience. In research, it provides categorization of balance concepts for analysis, helping to focus research questions and procedural system evaluations. For instance, by guiding and

restricting what aspects of balance are being evaluated. Or, by providing a clearly defined design space for defining how fitness functions and evaluation mechanisms can most accurately capture the type of balance being investigated. This is important for automated systems and research testbeds that involve evolutionary algorithms, whereby both the generation of an artifact and the ultimate evaluation are deeply dependent on how the researcher defined balance and translated it into a fitness function [433].

4.4.2 Multidimensionality of Balance

Balance can be sliced along multiple categories, but no one conceptualization should be assumed to operate wholly independently or be definitive of whether a game as a whole is readily perceived as balanced. A study might focus on game difficulty, but that might include tuning the economy of the game or tweaking the range of available player skills, or even tweaking the aesthetic properties of the game to make the player feel more or less powerful. Understanding that balance is multidimensional and can be sliced along multiple thematic categories and concepts informs how we can better meet player expectations and research/design goals. Specifically, the multidimensionality of game balance helps reflect the complex, varied, and detailed definitions provided by practitioner and design frameworks [138, 246, 420, 422, 33, 323, 117]. Although most authors do not have the same lists of balancing considerations, a summation of these approaches in the form of a taxonomy helps to capture all considerations that have been made across a larger corpus. The lack of generalization of balance suggests that implementing balance depends on the taxonomic dimension and balance context (e.g., target audience, player experience, etc.) of a designer before being carried out

or evaluated. Ultimately, this multidimensional approach means that we should avoid making sweeping generalizations about balance as a whole.

4.4.3 The Boundaries of Balance

Our taxonomy also helps us understand what balance is *not*. Although the methods of game balancing are extremely diverse, there is the anchoring concept of fairness throughout all categories. Is it fair to have more narrative than game, or vice versa? Is it fair to always lose to someone with more skill? Is it fair if only one style of gameplay is viable? Each subcategory of game balancing asks us to consider the game context explicitly through fairness, while other elements of design might look for other affects and phenomenal experience to target. Such discussions of fairness allow us to view balance not only in its subdivisions and multidimensionality, but also as a super category of game design that has its own texture, capable of telling stories about how fairness is given and pulled from us throughout our interactions with a game[1].

4.5 Limitations

A limitation of our work is that it was focused on academic publications and design framework publications. That being said, two more audiences could meaningfully flesh out the game balance and its perception and implementation: players and designers. While the taxonomic concepts and themes would not change, the distribution of codes might shed light on what players and designers focus on. Looking to industry

[1]Not to be confused with the subcategory of narrative feedback within our taxonomy—the narrative I refer to here is the sum of all aspects of balancing judgments through gameplay.

sources whose responsibility is to implement game balance in successful games can give us insight into how to polish a balancing design framework and provide further context into what design moves are made at what times for what reasons. Player perspectives are important because they fill out the audience identification theme, which might significantly differ from what researchers or designers think about balancing perspectives and what gaps might exist between designer intentions and player reception. While I have conducted interviews to qualitatively capture industry and player perspectives, a more systematic and wide ranging study of such views would bolster the conclusions presented in this taxonomy. All of that being said, the inclusion of 95 different sources in academic and pedagogical concepts provides a solid lens to start understanding balance across different contexts.

4.6 Conclusion

This chapter provides a centralized definition as well as taxonomic tooling to enable the application of that more general definition to specific sub-domains of balance. This approach allows me to quickly tell you what balance is about (fairness) while providing precise and exact direction on implementation strategies for each variation of balance we may discuss.

Chapter 5

Equalization: A Framework for Balance and its Proceduralization

> "...we want to encourage a variety of play styles. So if you're very PvP focused or if you're more exploration focused, if you're more stealth focused, we really want people to be able to play the game in a variety of different ways. But if the game is unbalanced, then one or more ways of playing could become untenable."
>
> Melody Ju, Embark Studios

After I had completed a Taxonomy and gone back to technical work, I had the realization that game balancing in the mode I had conceptualized (considering goals,

audience, context, game elements, algorithmic procedures) could be applied to every balancing operation—without exception. My belief in frameworks held strong—that, when designing, conceptual and heuristic frameworks allow us to get creatively unstuck while retaining our designerly intent. As such, I returned to the data set put forward in the previous chapter and set out to concretize the framework that I had in my mind, so that all future approaches I took to balancing systems had a clear and reproducible pattern. With a framework, I could always have the next step when I struggled to find the proper solution!

In the last chapter, I covered how the usage of balance in research work can be split into an array of definitions surrounding balance, definitions that are sometimes at odds with one another. The complexity and multidimensional nature of game balance means that a game or system designer responsible for a game's balance carries a heavy burden. They are responsible for hypothesizing, implementing, and testing a loosely bundled set of emergent properties that do not seem to obey a singular, specific definition across audiences or game contexts. This challenge is amplified by the fact that game designers must encode systems of balance in code and rules—a process that asks the near-impossible task of translating and predicting player sentiment and baking those reactions into the core of a game itself. This process of systematizing, testing, and tuning occurs throughout nearly every step of the game development process, extending even past a game's initial release in the form of patches and updates [247]. The importance, frequency, and difficulty of balancing a game raises the question: how do we decide how to implement it for any given game?

This chapter presents a generic design framework for game balancing that I call

"equalization". I don't mean that a game's balance means that perfect equilibrium, as storied game designers (such as Masahiro Sakurai [349] or Alexander Brazie [47]) have noted that a "perfectly balanced" game usually is not a good experience, but having a dramatic tension around nearly losing or nearly winning combined with the meta-challenge of identifying losing strategies and mastering winning strategies is key to a game that is perceived as balanced [157]. Instead, I use the term equalization the way it is used in music production or when building climbing anchors. In music production, different voices of the song are tweaked to match the expectations of an audience. In a dance song, the producer might bring forward drum and bass tracks, emphasizing rhythm for an audience at a dance hall. In a jazz song, the producer might instead focus on the voices of the soloists, allowing an audience to appreciate the talent and beauty of the performers and their instruments [355]. In climbing, equalizing an anchor (a setup that holds a rope in place for a climber) means anticipating the way the rope will pull when weighted such that individual elements of the anchor remain safely in place if the rope is suddenly submitted to the force of a falling climber [326].

In both of these examples, there isn't a one-size-fits-all approach to setting up the perfect song or anchor—it takes a thorough reading of the audience and environment combined with a technical skill that reflects and anticipates those external factors. That being said, there are generic implementation strategies for building the song or anchor such that it satisfies its core responsibilities. Similarly, in balancing, there is no singular, well-defined algorithm that will solve every gameplay issue. Instead, there are the challenges of scoping, selecting an implementation strategy, and checking if that strategy worked. As such, we present a high-level framework that helps designers start from a

blank slate when working on a game's balance, providing a catalog of implementation *methods* and game system *targets* that are viable for a designer's desired *goals* and *audience*. I have derive this framework by extending the prior chapter's inductive thematic analysis over 95 publications, pulling key terms related to game balancing methods and consolidating them into a subset of ways to balance different types of game elements and systems. I supplement this by providing framing around how a designer defines their goals, game context, and audience, followed by a critical evaluation step that helps a designer understand if a balancing strategy works. We define this framework as iterative and open loop—it occurs in many cycles during development, and can be started at any step of the loop. This framework helps to provide a template for designers to define the what, why, and how of game balancing, regardless of the context of the audience or design goals of a game.

5.1 Background

Before diving into the framework proper, it is important to understand why frameworks are important. Design frameworks are useful tools that help designers make an ill-defined problem space with a large variety of potential solutions tractable. Frameworks are useful for handling "wicked" or "cursed" problems in game design, where clear design goals might exist but the path to reach them is hazy [401, 216]. Frameworks provide structure and heuristics for navigating a design space, providing the ability to know what to do and when to stop during the process of solving a design problem [156]. When there is a large amount of uncertainty in a problem space, understanding what

methods can be used at what time to resolve that uncertainty becomes a pivotal skill to move closer to an ideal solution [75]. Using frameworks to navigate design and problem spaces provides major time and cost savings—eliminating unnecessary prototypes, dead-end design directions, and uninformative playtesting cycles can save studios and researchers time and labor when trying to design a game. Frameworks not only allow us to move to a design solution, but also help us "debug" when something feels wrong in a game by teaching us the right questions to ask in response to a problem.

Game design in particular has a storied history of design frameworks across a wide variety of elements of the game design lifecycle. The Mechanics, Dynamics, Aesthetics (MDA) framework of game design is one of the most prominent in game design literature, which argues that the end-player aesthetic experience is generated from the run-time dynamics of a game, which in turn are produced by the designer-defined mechanics of a game [199]. The General Video Game AI framework seeks to provide generic and well-patterned ways to test artificial intelligence in a wide variety of game contexts, separating the evaluating gameplay agent from the game environment [367]. Frameworks and their inspiring framing have also been explicitly incorporated into games research, especially when it relates to procedural content generation, as the frameworks can be in turn encoded inside of game content creation algorithms. Smith et al. [456] do so by providing a framing for 2D platforming games through the lens of musical theory, using the metaphors of rhythm and beats to define what attributes of a 2D platforming game level are important to the feeling of gameplay, then implementing it into a level generation system. The metaphor of "beats" is also used by Mateas and Stern [303] in the context of a narrative framework that aims to decouple narrative into stories and

behavior pieces that inform the dramatic performance of a narrative plot. The utility of frameworks in game design, both at a global design level and an explicit implementation level, indicates that the existence of frameworks that tackle critical design challenges not only guides designers generally but also enable the creation of novel game systems and player interactions.

The importance of frameworks also belies another advantage over a more explicitly implemented strategy: they are generic enough to handle a wide set of cases while still providing solid stepping stones to achieving a design goal. This dissertation contains two experimental strategies (Chapters 8, 9) that do demonstrate opinionated implementations, but the importance of their documentation demonstrates how this framework can get a designer or researcher from nothing to a focused, clear final product.

5.2 Methodology

The methodology of this work extends the prior chapters, so details around data collection can be cross-applied here. The primary difference in this chapter is that instead of focusing on the definitions of balance, we instead focused on the process of balancing, highlighting common methods and approaches to the problem (especially as it relates to procedural concerns).

5.2.1 Inductive Thematic Analysis

An inductive thematic analysis [413] was performed on the 95 publications collected, focusing on collecting codes where game balance definitions, implementation

methods and evaluation strategies were specified. The team started by performing an inductive coding process, collecting key terms related to game balancing procedures (e.g. "playtesting", "genetic algorithm", "weapons", "damage rate"). Each publication had a corresponding set of codes. If a single paper contained the same code multiple times, that code was only counted once for that publication. This resulted in a list of 430 unique codes with a total of 783 occurrences in the literature. From these codes, the team found that the 430 codes landed in one or more of the following categories:

1. **Goals** (264 Codes, 454 Occurrences): Describes the audience specification, process of setting designer goals and framing tools, and the evaluation of a completed system.

2. **Targets** (275 Codes, 322 Occurrences): Describes specific elements of a game that can be used when tuning game balance. Examples include levels, classes, and NPC behavior.

3. **Methods** (340 Codes, 636 Occurrences): Describes methods of tuning balance given a target and goal. Examples include simulation, deep learning, and playtesting.

Some codes could be considered to belong to more than one category. For example, "asymmetric roles" could be a method (introducing asymmetry in roles could be used to tune up or down an approach) or a target (the roles themselves). When this happened, we included the code in each category it could belong in. We used this approach to allow flexibility in interpretation for each of these terms, as well as to

capture the multidimensionality of how these terms are used in different elements of the game balancing process.

From our final list of codes, a round of clustering was performed using axial coding [84, 329] to consolidate similar or related codes into thematic categories under our primary three categories. For example, "cost curves" and "economy generation" were grouped under the method subcategory of "economic modeling". In the following section, we break down these thematic categories.

5.3 Results

The literature review produced 5 subcategories of balancing goals, 10 subcategories of balancing targets, and 12 subcategories of balancing methods. While I do not view the following list of goals, targets, and methods as exhaustive of all game balancing element approaches, I think the thematic similarities across the research literature demonstrates a representative sampling of critical aspects of any game balancing process. Below, I provide a short description of the larger themes and provide the subcategories of each. A link to our data used in this review can be found in appendix B.2. An overview of how codes are related to each other is shown in figure 5.1.

5.3.1 Goals

Balancing *goals* refer to the framing techniques that designers and researchers can use to mitigate scope concerns and identify how exactly they want their game to be balanced, and what path they should take to identify their audience and success criteria. We identified 5 subcategories that consist of the formulation of balance problems by

Figure 5.1: A diagram showing the three major categories of codes found. The top three terms are shown as an example in each individual category, and examples of overlapping categories are shown where the major categories overlap. Major categories also show the number of unique codes and overall frequency of code appearance.

setting designer goals, specifying an audience, and evaluating how the system worked. These are labeled goals as they represent the design direction and evaluation of any strategy used to balance a game. Table 5.1 shows a breakdown of designer goals in game balancing.

Table 5.1: Goals Subcategories and Descriptions

Goal Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Audience	84	Player Profiles, Mavericks, Personality Traits	The desired audience that the designer wants to engage with their game.
Balancing Goals	60	Replayability, Winnable, Diversity of Gameplay Experiences	The goals a designer sets for a successfully balanced game or system.

Continued on next page...

Table 5.1 (Continued)

Goal Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Designer Framing	98	Design Dimensions, Intention Alignment, Luck and Skill	The set of strategies, definitions, and design spaces that are available to a designer while balancing a game.
Audience Outcomes	85	Anxiety, Engagement, Player Self-Esteem	The potential outcomes the game can have on the specified audience.
Evaluation	127	Win Rates, Experience-Sampling, Believability	The methods used to evaluate if a balancing goal was met.

5.3.2 Targets

Balancing *targets* refer to the game elements that can be tuned to affect game balance. These cover the systems, characters, controls, and the like with which the player interacts. We identified 10 dominant categories for targets, ranging from simple

game metrics (e.g., health) to complex game systems (e.g., character builds). Targets suggest relevant parameters and attributes that can be tuned to impact the balance of the game. For example, the level of "bullet magnetism" (how much a bullet will stray from a player's reticle to hit an enemy) can be increased or decreased to impact how difficult it is for a player to hit enemies in a shooting game. Table 5.2 shows a breakdown of these subcategories below.

Table 5.2: Targets Subcategories and Descriptions

Target Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Game Environment	34	Enemy Volume, Obstacles, Item Spawning	The surrounding game world in which play occurs. Includes things like traversal and distribution of other game elements.
Game Elements	44	Units, Vehicles, Equipment	Game objects that the player interacts with as part of gameplay.

Continued on next page...

Table 5.2 (Continued)

Target Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Player Actions	29	Abilities, Jumping, Running Speed	Actions available for the player to perform and their corresponding attributes.
Mechanics	64	Luck, Gravity, Time Limits	Interacting game systems that define interaction between other game elements.
Economy	40	Drain, Market, Scarcity	Economies within game involving how resources are distributed throughout the game world.
Non-Player Characters	18	Enemies, AI Behavior, Multi-Agent Systems	Game elements other than the player that represent system/AI-driven characters.

Continued on next page...

Table 5.2 (Continued)

Target Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Match-making	18	Rankings, Asymmetric Roles, Community	The composition of players and their teams.
Information	20	Hidden Information, HUD Restrictions, Game Map	Information critical to gameplay that is presented or hidden from the player
Progression Systems	28	Experience, Power Curve, Scoring System	Systems related to player progress and pacing within the game.
Player Controls	27	Aim Assist, Steering, Reaction Time	Methods related to direct player control of the game.

5.3.3 Methods

Balancing *methods* refer to the processes that a designer can use to tune *targets* in pursuit of game balance. These cover algorithmic, manual, and analysis techniques to understand and change the state of game elements and systems to get a varied balancing output. We identified 12 subcategories of game balancing methods, including using reinforcement learning approaches, playtesting cycles, and implementing runtime balancing systems. Methods suggest explicit approaches to tuning that can be used during design, runtime, or through post-release balancing patches. A learning system might be used to discover the correct parameters for a combat system prior to release, but another learning approach might be used at runtime to adapt to player behaviors; it might also be applied to aggregated player data to inform a design team on how to tune a game after it has been played publicly. As such, methods aren't necessarily unique to any game target, but represent approaches that can be applied in a large variety of contexts. Table 5.3 shows the breakdown of the balancing methods.

Table 5.3: Methods Subcategories and Descriptions

Method Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Runtime Adjustment	63	Balancing Agent, Dynamic Difficulty Adjustment, NPC Parameter Scaling	Modification of the game state at runtime to meet a player's balancing needs.
Information Visibility	14	Advance Warning, Longer Tutorial, Artificial Gating	Control of the information available to the player which in turn informs their future actions.
Simulation	74	Automated Playtesting, Simulation-Driven Balancing, Rule Generation	Automation of gameplay in order to inform balancing strategies.

Continued on next page...

Table 5.3 (Continued)

Method Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Machine/ Reinforcement Learning	84	Artificial Neural Network, Q-Learning, Supervised Learning	Application of learning strategies to gameplay tuning or game element behaviors.
Analytics	50	Data Visualization, Player Modeling, Gameplay Data	Usage of collected data to understand how games are played and the characteristics of player personas.
Design/ Playtesting Loops	66	Playtesting, Iteration, User Evaluation	Loops of playtesting and manual tuning between players and designers.
Parameter Tuning	54	Buffs, Nerfs, Hand Tuning	Identification and tuning of critical parameters in a game system.

Continued on next page...

Table 5.3 (Continued)

Method Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Logic Programming	5	Domain-Specific Languages, Provenance, Graph Grammars	Usage of logical programming to find balanced game states.
Search-Based	80	Evolutionary Algorithm, Monte Carlo Tree Search, Hill-Climbing	Search strategies to find a balanced game state.
Economic Modeling	19	Economy Generation, Resources Transition, Trade-Offs	Usage of economic analysis to find a ideal game progression.

Continued on next page...

Table 5.3 (Continued)

Method Subcategory	**Number of Code Occurrences**	**Example Codes**	**Description**
Game System Interactions	27	Functional Linkage, Balancing Architecture, Effect Dependencies	Analysis of interacting game systems for impacts between different states of balance.
Player Skill Determination	100	ELO, Rubber Banding, Matchmaking	Methods to identify player skill and tune the game state or multiplayer environment to meet those player skill levels.

5.4 Game Balance Equalization Framework

Given the above coding, we have three main categories of language used when describing game balancing practices. That is, one pertains to how we frame and set goals for our game balancing processes. The next deals with what parts of our game we want to tweak in order to meet the aforementioned goals. Then we have a wide catalog of methods that we can use to deal with those targets in pursuit of our goals. With

these elements in mind, we can return to our metaphor of "equalization"—how do we focus on and move between these different aspects of game balancing so that we get a song that meets the needs of its audience and environment?

This framing from the literature informs key requirements for a game balancing framework: a designer must have a goal and audience in mind, they must understand the context that the game will be played in, and they must develop an approach to targeting and tuning game elements. Finally, the designer must test their established goals and be ready to repeat the cycle until the balancing feels just right. With this in mind, we can construct a framework for game balancing, seen in figure 5.2. Each node can be viewed as a way of "equalizing" or focusing on different attributes of balance, letting us focus on a desired outcome, the targets we leverage, the methods we use, and so on. The following sections will go over every major node in the framework, detailing key considerations and approaches when using the framework.

5.4.1 Goal Hypothesis

When attempting to balance a game, it is important for the designer to have an ideal end-state in mind relating to how the game's balance is perceived by the player. Many works detail the construction of competitively balanced games in which players feel rewarded for their skill and mastery [409, 186, 117]. This task may be somewhat trivial in symmetric games such as Chess, but when considering the more common asymmetric forms games take (different characters, abilities, etc.), tuning a complex system of characters is a high bar, and the designer should have a hypothesis about what experience their game's balance is driving in the player. In a game like *Dark*

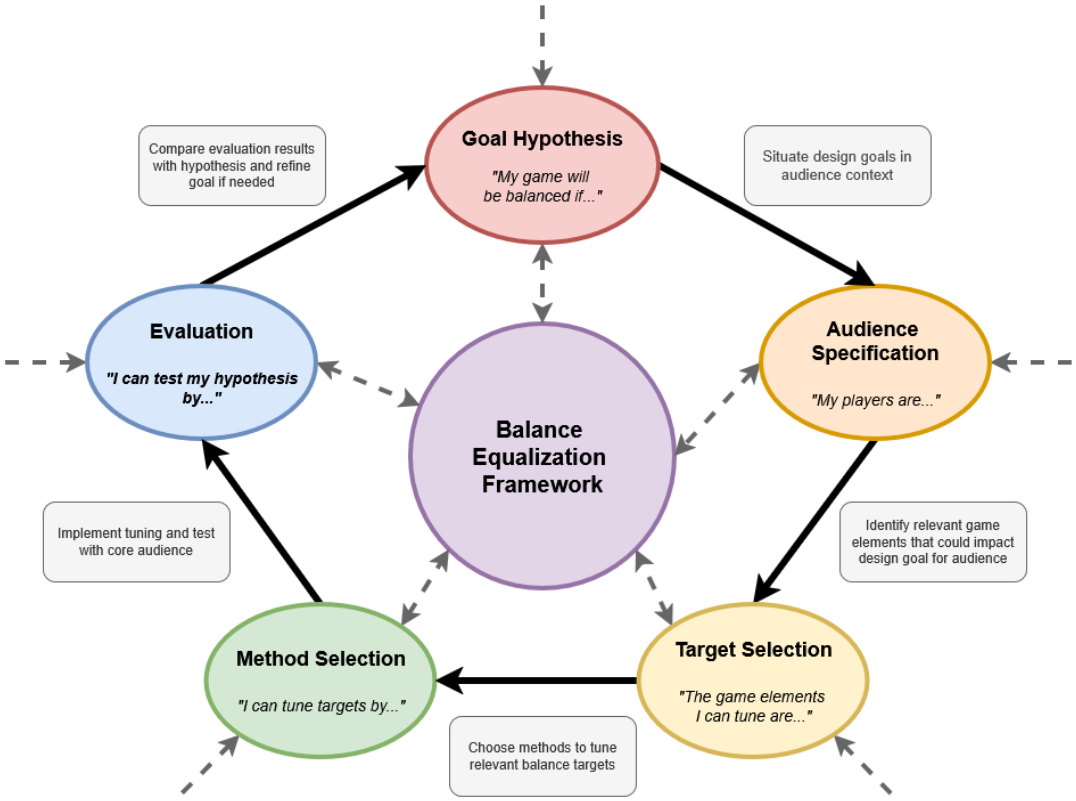

Figure 5.2: A visual representation of the equalization balancing framework. It forms an iterative loop, asking designers and researchers to understand their goals and audience, identify game element targets, methods to tune said targets, and strategies to evaluate the system. The framework can be entered at any step, and is intended to be looped through until a balanced game state is identified.

Souls [135], that hypothesis might be one of continual struggle until a glorious success; in something like *Mario Kart* [114], it might be always having the hope of coming back from last place. Although this goal can take many forms, it should be kept in mind throughout the rest of the balancing process, as it provides rails on how to make decisions on all future steps.

5.4.2 Audience Specification

With a goal in mind, a designer must start questioning context, starting with the question: who will be playing my game? In the prior example of competitive games, the designer might have an age range of players with prior interest and expertise in the genre they are designing within. Doing so enables the designer to research similar games and communities, finding failures and successes over the history of games. In competitive games, a designer might look at the warm competitive reception of *Super Smash Bros. Melee* [173] and the icy reception of its sequel *Super Smash Bros. Brawl* [457], which was so poorly regarded by the competitive community that they took it upon themselves to modify and rewrite elements of the game engine, character moves and timing, and even network connectivity in order to finally get a game they felt was balanced (the modded game is known as *Project M*, referencing a desire to have the sequel perform closer to its predecessor) [147, 354]. As such, knowing the audience that will primarily be playing (and buying) the game lets the designer continue to build out the rail laid down by their goals. It is not just that the new fighting game should reward competitive play, but players of our games enjoy responsive combat, the ability to chain combos together, and the discovery of unexpected advanced techniques

that provide an incredibly high skill ceiling. Tools such as building player personas [30] and researching player motivations [98] are critical in this step, as they make the goals specific and testable with an audience, where testing with too wide an audience would introduce unwanted noise into the evaluation process. Such insights about the audience further narrow down what targets and methods will eventually be selected as balancing candidates.

5.4.3 Target Selection

With a goal and audience in mind, the designer must now select which parts of the game are going to impact balance in the way that they desire. Continuing our competitive game development, if our players want responsive combat and the ability to perform combos, the designer would start to consider the game elements that have the greatest impact on those outcomes. This might be character speed (enabling faster combat), move wind-ups and hit-stuns (enabling chaining of attacks together), and a permissive physics system (allowing unexpected combinations of inputs to produce unique movement patterns). These targets form the core of what in the game state the designer will be targeting. In many cases, especially when using more procedural or algorithmic balancing methods, the parameterization of these elements proves eminently useful, as their translation into numeric or enumerated values allows them to be searched and tuned quickly without major architecture changes. Based on our sampling of codes, there is a wide variety of targets available for tuning. In an adventure game, a designer might tweak obstacles in the game environment, the economy of potions and weapons at shops, the information provided by maps and NPCs, or even the behaviors and skill

of NPCs themselves. In any case, using the goals and audience can help narrow down what targets are viable for tuning, which aids in building out the method the designer will use to ultimately balance the game.

5.4.4 Method Selection

At this point, the designer should have a design goal, players with whom they want to engage, and viable game elements that could impact their design goal. From here a variety of manual and automated strategies can be employed to achieve a balanced game state. Perhaps the most tried-and-true method is the playtest-and-tweak loop, where a build of the game is played by designers or playtesters, reviewed quantitatively and qualitatively, and then tuned for the next round of playtesting. While this is undoubtedly effective, it is a time-consuming endeavor. As such, many of the methods found during our code consolidation provide methods to automate or optimize elements of game balancing processes, using a wide variety of approaches. Search strategies can be useful for identifying optimal parameter tunings across a wide set of tunable targets [295], and NPC behaviors can be tuned to match different levels of player skill through reinforcement learning [15]. Once the methods are applied to the targets of balancing, the game becomes ready for evaluation, which provides the framing for the designer to finally evaluate if their goals are being met by their balancing process.

5.4.5 Evaluation

At this point, the designer should have formed a goal, targeted an audience, identified key game elements to target, and implemented some form of balancing through

their desired methods. Evaluation comes in many forms—human playtesting is the ultimate goal, but automated playtesting can also provide preliminary signals on if the balancing approach taken is yielding results. Quantitative and qualitative data should both be taken into account during the evaluation. The competitive game example we have been using might look at win rates of various characters and strategies as a success indicator through quantitative analytics; but it should also ask the players themselves if they found the sense of competition rewarding, engaging, and diversified. Tools can help shine a light on how effectively games are being balanced—reviews of playtraces [436] and visualizations of class/strategy usage [370] can quickly give insight into the invisible traits of game balancing. Playtest procedures such as "think aloud" protocol [190], gameplay video reviews [205], and player surveys [165] can help to understand how the player experiences the balance of a game. Crucially, these two domains should be mixed—if the quantitative signals selected for evaluation are not lining up with the qualitative observations from a desired audience, the quantitative strategy for balance analysis likely needs to be reworked.

5.5 Discussion

5.5.1 Using the Framework

The framework provided above can be read linearly, but the reality of game development means that nodes might be visited out of order or require multiple visits to reach balancing goals. Designer goals might be set early on, but playtesting could reveal that different elements of the game are more worth the designer's time, necessitating a

reset of their goals. Likewise, a reinforcement learning method to tuning might over-optimize and produce boring gameplay, requiring designers to replace that balancing strategy with another approach that produces more nuanced outcomes for their players. Through this, while the framework has a suggested ordering, it should be understood that at any point the designer might need to return to a given node in pursuit of a balanced game. Furthering this point, it is very rarely the case that a single traversal of the loop will produce a balanced outcome—the goal of the framework is not to produce a perfect outcome on the first try, but to provide a heuristic to select which steps of balancing should be returned to and tweaked until a designer goal is finally met. The loop may be traversed dozens of times—but it allows a designer to approach balance in a focused and narrowed manner, avoiding "flailing" while trying to figure out what key change will make their game more balanced.

The codes provided in the literature review section of this paper provide a brief, non-exhaustive catalog of approaches that can be used during game balancing. The variety present in the codes demonstrates that not only are goals around balancing multidimensional, but the strategies to make a game balanced are as well. A designer might balance competitively, but they could also balance a game for a casual, cozy experience that rewards slow and relaxed gameplay over short, frequent gameplay sessions. Both of these game contexts are wildly different but might still have overlap in the targets and methods that can be used to tune them. On the other hand, two competitive games in the same genre might target differ greatly in their balancing targets and methods—one might look at tuning each individual character, while another might focus on the matchmaking ecosystem to make a game feel more balanced. In both of

Goal	Adapt to Player Skill Levels	Allow New Players to Have Success	Make Navigation Challenging	Make Game Rewards Scarce	Confirm Balance in Competitive Scene
Targets	Agents	Player Control		Economy	Metagame
					
Methods	Reinforcement Learning	Aim Assist	Information Telegraphing	Reward Curve	Gameplay Data Analysis

Figure 5.3: A table demonstrating the balancing framework across a variety of use cases. Starting from top to bottom, each column shows a design goal, a game target, and a balancing method. The diversity of cases presented shows how the framework provided is adaptable for the wide array of game balancing definitions.

these examples, the framework provided here is still useful—it provides rails and narrows decisions when considering which approach to take, where. Figure 5.3 shows a set of examples that consider different goals, audiences, targets, and methods during the game balancing process, showing how the framework can be applied in a wide variety of contexts.

5.5.2 Turning Framework into Procedural Architecture

Throughout the coding process and the results presented above, a pattern emerged concerning the framing of balancing architectures: static and dynamic balancing systems. While both are valid and can be used within our design framework, they provide an interesting note on when balancing can be applied. Static balancing refers to the balancing of game targets prior to runtime, meaning that the game balance is set in stone when it is released. Dynamic balancing refers to the balancing of game targets using balancing methods during the runtime of the game, producing an experience that

is adapting a sense of balance the more a player engages with it.

Both of these approaches are valid and have produced successful games—*Left 4 Dead* [498] and *Resident Evil 4* [498] both employ dynamic systems to great success. *Left 4 Dead* uses an "AI Director" strategy to tune the intensity of play over time, while *Resident Evil 4* tunes up or down the difficulty of enemies based on how many times a player has died. On the other end of the spectrum, games like *Super Smash Bros. Melee* have been viewed as competitively balanced games for over 20 years without ever receiving a balance patch or containing runtime systems that modify gameplay to tune game balance. What is interesting about the framework when placed in procedural contexts is that it codifies design heuristics into software systems—a game system must hypothesize what the player wants, understand who the player is and what their current state of engagement is, decide what to modify, how to modify it, and then check if their changes meaningfully changed the experience. In this manner, dynamic game balancing approaches can use the framework to make explicit procedural systems for game balancing in code and game systems. The AI Director pattern is an excellent example of this—it has a goal (make the game more or less intense), evaluates its audience (what does the game look like and what is my player doing), picks targets (enemy spawns), employs a method (increase or decrease spawns in strategic locations), then returns to see if its goals have been met before repeating this process.

Going a step further, static game balancing techniques can be applied to dynamic game balancing systems prior to runtime. A dynamic system still has static parameterizations (how frequently do I change the game state, to what degree do I change the game state) that need to be tuned prior to gameplay. As such, the design

framework provided above can be used to develop a static game balancing approach to tune dynamic game balancing systems. The design of such hybrid architectures already implicitly exists, but applying the design framework can help focus down how tuning and evaluation of dynamic systems can be done effectively.

These insights of applying the framework presented here in procedural contexts form the foundation of how I attempt to increase the efficiency and quality of balancing systems through my technical design exercises in part three of this work. It also sheds light on how procedural methods can open up new design spaces when considering how balance tuning can lead to emergent outcomes that are still fair to players. Namely, taking the framework, framing audience and experience goals, implementing automated tuning and playtesting strategies, and going from there forms the ground-floor of how we should frame and build automated systems.

5.6 Limitations

As noted throughout this chapter, we cannot possibly provide an exhaustive list of balancing approaches—we can only summarize the approaches that have been published in the literature. This also potentially biases our results—the focus on academic literature means we are likely over-indexing methods viewed to be novel and experimental instead of well-established or reliable. In addition, many approaches are hybrid amalgamations of methods and targets, making the identification of every permutation an impossible endeavor. The academic focus on novel experimentation and the use of hybrid methods might explain the heavy prevalence of machine learning and

reinforcement learning in the context of game balancing. On the flip side, our review suggests under-explored areas of game balancing that might be ripe for exploration. The low incidence of logical programming approaches to game balancing indicates that it is a high-potential area of research. Luckily, industry research presented in chapter 2 alongside designer interviews peppered throughout this dissertation highlights well-established methods. Ian Schreiber, for example, highlights the usage of spreadsheets and mathematical models in early 2000's balancing efforts, while all interviewed designers noted the importance of playtest loops alongside qualitative observation for tuning.

With the acknowledgment that all balancing targets and approaches cannot be cataloged, it is still useful to review the examples of methods that we have available alongside the research that has promoted them.

Chapter 6

Catalog of Targets and Methods for Procedural Balancing

> "[Dynamic NPC memory of player actions] forces players to come to terms with certain habits they have... I really want to stress the importance of this single player ability to train in a realistic environment if the game is aiming to have a competitive environment."
>
> Dominick Reba, Project M/Project+ Contributor

We have a framework now, and I noted that balancing procedures can be applied to game *targets* using balancing *methods*. I gave some categories, but if I

am trying to make a procedural balancing system, what is actually available? This chapter gives a (non-exhaustive) catalog of game targets and methods according to the prior chapter's categories that can be used as starting points and were considered for my technical work during my degree. This catalog will not go into extreme detail on any of these methods, but will provide enough to chase down the source material and tinker yourself. This chapter serves to be the explicit paths I can apply inside of my framework—the catalog of strategies I wish I had when I started my dive into the world of systems design and balancing.

6.1 Static, Dynamic, and Hybrid Balancing Strategies

Before diving into any specific strategy, it is necessary to consider global strategies in game balancing: static, dynamic, and hybrid. These cover the "when" of when balancing occurs, in both manual and procedural balancing methods. Static balancing is purely design time—it is game balancing done during game development that does not change while the game is being played. Creating a defined curve of enemy difficulty, setting unchanging attributes of characters, setting up spawn points—all of these would be defined by the designer and left alone once a player boots the game up. On the other hand, dynamic game balancing refers to balancing strategies that occur during runtime as a result of player input [32]. Examples include changing enemy difficulty in response to the player's character level, dynamically tuning the level of aim-assist on a player's weapon, or mechanics that help lesser-skilled players reasonably compete with experts. A prominent area of dynamic game balancing research is called Dynamic

Difficulty Adjustment (DDA). DDA is an approach to dynamic game balancing that focuses on adjusting the difficulty of a game in response to the player and game state during runtime [198]. DDA is a heavily researched area of game balancing [546], driven in part by its ubiquity and diversity of methods in industry. An academic investigation of its use in industry at *EA Games* is documented by [530]. DDA can take many forms depending on the design goals of the game. For example, a DDA system in the game series *Mario Party* [196] provides unexpected rewards to players based on gameplay metrics outside of ranking considerations, providing opportunities to include players with variable skills that reasonably compete in a single game environment [503]. On the other end of the spectrum, DDA might seek to change the system to meet the player's current level of skill and/or gradually increase it over time to keep a player in a "flow" state (a state of challenge that keeps the player from being too bored from a lack of challenge and too frustrated from too great of a challenge) [30, 262]. An example of this is in the game *Homeworld* [444], where the volume of enemy spawns is increased if a player is successful in prior scenarios. A third design direction for DDA systems is the adherence to a desired dramatic pacing, such as in the game *Left 4 Dead* [498], where an AI director seeks to provide a moderation of perceived player tension such that there are alternating periods of increasing and decreasing difficulty, mimicking a dramatic curve [45].

Static and dynamic balancing are not necessarily completely different schools of design. Hybrid architectures of these approaches exist and are sometimes required, particularly in the case of dynamic balancing. The aspects of dynamic balancing that determine things like the scale of balancing change, change frequency, and so on ultimately

have a starting point that are statically defined. Tuning *how* the dynamic system works still requires the designer to define policies, scalars, and parameters at design time. On the other end, dynamic systems might be used to discover optimal static tunings prior to play. Evolutionary strategies in static tuning reflect this, especially when automated playtests driven by reinforcement-learning are involved [409, 268, 323]. In this regard, we should acknowledge static and dynamic balancing as useful conceptual tools, but in reality we will often merge such approaches to get a true, customized system for our game design needs.

6.2 Balance Strategy Catalog

The following catalog is intended to show the combinatoric range combining balancing targets with methods. The idea is that each target should help point to an area of a game that *can* be balanced, and the methods point to *how* you can balance said area of the game. Any target can be combined with any method, which gives this catalog a "build-a-bear" usage. One should not feel restricted to one-target or one-method, but can combine them in multiples depending on the design and balancing goals. Although these categories were drawn from a literature review, they should not be taken as exhaustive: games are a wildly complicated and diverse design space. However, the framing of these categories should also provide patterns for a designer or researcher to identify and group novel targets and methods they encounter.

6.2.1 Targets

Below are the targets discussed in the last chapter for game balancing, with examples illustrated for each. These categories should not be taken as isolated, complete indexings of the what makes a game a game. Instead, they are rough units that can be tinkered with during balancing to achieve goals. The following are the useful categories we can use to focus on a balancing methodology or determine how different parts of a balancing system overlap with one another. Some categories can encapsulate elements of others (game elements, for example, certainly overlaps with game economy), but it is still true that they have independent lenses that can be used for balancing purposes. A final note—in order to manipulate these targets in the methods section, they should be conceptualized, they need to be thought of in terms of parameterization. The translation of conceptual game targets into computer-readable interfaces enables balancing in manual contexts and is critical for balancing in procedural contexts [430]

Game Environment The game environment category of targets refers to how the external game world presents obstacles and opportunities to players. In the context of balance, examples of environment tuning relate to the ability of a player to traverse [387], the spawning points of players, items and enemies [198, 137], or the positioning of design elements to drive different elements of pacing [197]. While easiest to conceptualize in terms of games that involve a player-character moving through space, puzzle and narrative games can also have their environment tuned by focusing on how meta-level progression is navigated by the player [254].

Game Elements Game elements is a catch-all term for objects related to gameplay that are generally parametrically defined. In a card game, this might be the cards [101]; in an action RPG, it might be a player's health points [26]. Inventory items (and their prevalence) also relate to this, as they confer boons on players based on their attributes and quantity [379]. External elements like generic units, characters, vehicles, and so on all help round out the game, and control over their frequency and characteristics can drive the player's progression in the game.

Player Actions Player actions are the "verbs" of gameplay, the available interactions and their characteristics that enable the gameplay writ large. The base actions (attacking, moving [342]) are here, as well as the framing around these actions (cooldown, damage rate/scaling [186]). These are a common and useful focus for balancing—they are immediately felt by the player in terms of game dynamics, and have an out-sized impact on what the player can do in the game world.

Mechanics The downstream interaction of actions, elements, and environments leads to mechanics. This might be the durability of inventory items [101], the setting of players against a horde of NPCs [322], randomness rolls in the system [217], or time limits during a level [7]. I will note—the debate of distinction between game elements, environment, and mechanics is an extensive one and not necessary to resolve for this discussion on balancing—a gut feeling on what belongs where should be enough to move forward with tuning. Changing the properties of how a mechanic operates and the characteristics of its constituent elements are all valid approaches to balancing.

Economy Game economy refers to game systems that are near direct metaphors or implementations of economic structures. This might refer to abundance or scarcity in a game economy [198], the costs of purchasing, using, or trading game elements [379], the presence of faucets/drains to control total capital in an economy [407], or the availability of various game units through its economy [348]. If one is able to game an economy too easily, the player can bypass other balancing restrictions and negate other meaningful decisions in the game [126]. Likewise, too harsh of an economy can prevent player progress altogether—in many ways, having a working game economy is a prerequisite for maintaining many facets of a game's balance.

Non-Player Characters Non-player characters influence balance in two primary ways. First, their base characteristics can drive their perceived difficulty. If an enemy has too high of a health or damage stat, players might feel like they are boring "sponges"; too low, and the game becomes trivially easy [426]. Second, the behavior of an NPC agent also drives difficulty [217, 15]. If an agent responds meaningfully to player input, they become a welcome ally or worthy opponent. If they run into walls aimlessly, stand still instead of avoiding obvious danger, or otherwise behave unrealistically, not only can the game's sense of difficulty drop, but the overall believability and immersion can break for the player as well [520].

Player Controls Unlike player actions, player controls relate to the dexterity elements of the play driven by a control scheme. During gameplay, players may have to quickly aim a reticle, manage difficult turns, or react to sudden quick-time events. In all of these cases, the control interface between the player, their dexterity, and the game

system can be tuned to assist (or, in the case of "rage" games like *Get to Work* [213], frustrate) the player. As such, balancing can be used to manage players at different levels of dexterity. One can apply aim assists or bullet magnetism to help ease the challenges of managing a reticle [503]. Steering can be assisted, leading players to the correct turn path on a sharp turn [73, 531]. *Celeste* presents an excellent example of this style of tuning, offering a plethora of game options, such as slowing down time to make an otherwise punishing game available to a wide majority of players [52].

Player Teams/Metagame How players are paired with each other determines their perceptions of balance. This, along with asymmetry in games, presents opportunities to help define how team compositions and matchmaking strategies define the balance of the game [338, 98]. How games choose to measure and apply skill, force combinations of strategies, handle draws, and compute rankings all play into how balanced a multiplayer environment is felt [245]. As such, the metagame around how matchmaking occurs and what kinds of strategy are most dominant is based in some part on what types of player someone encounters online. In an extreme example, if a new player is constantly matched with professional players, that new player will likely feel like the game and the multiplayer is profoundly unfair. Balancing how matchmaking works thus impacts overall fairness perception of the game.

Game Aesthetics/Information How a player judges their abilities, the level and their enemies can dictate play. Similarly, a player acts on what they know about the game. As such, the availability of information can also influence the perceived and actual balance of the game. This information can be driven by aesthetics (how strong does that

attack look?) or game-system knowledge (what is this marker on my map?) [68, 203]. What the heads-up display tells a player, what is privileged information, how a community discovers innate information about the game all play a role in how information drives game balance. Aesthetics do as well—changing telegraphing animations of enemies and attacks may ratchet up difficulty [385], sound cues might help players identify correct actions [275] and removing cues of hidden objects make them harder (or impossible) to discover [106]. Think of this category this way—how does the balance of a game change when you have a game guide open next to your controller?

Progression Systems A category that crosses over into multiple other categories, progression systems are the overarching game systems that determine the development of the game over time. These can be directly tied to game elements such as stat-gains or character builds [544], difficulty spikes in level progression [282], or the intended pacing and sequencing of the game [30]. In all cases, progression systems form meta-groups of other targets throughout the game. Balance might need to change dramatically depending on player progression: The endgame (where a player encounters challenges and tasks after technical game completion) generally has a wildly different level of balance than the rest of the game [369, 547].

6.2.2 Methods

With targets covered, I turn to looking at the methods that we can use in order to achieve a balanced game. The methods listed below cover manual and automated balancing methods. These methods are target agnostic—the goal of this section of the

catalog is to provide a manipulation strategy of whatever targets are identified as key to game's balance.

As such, a conceptual architecture diagram is presented at the start of each method described (aside from information telegraphing, which will be explained in that section). These diagrams aim to provide a high-level, unopinionated understanding of these balancing processes without dictating the specific software architecture used when approaching them. I take this approach because the development patterns used for any of the below methods have many structures and approaches that will work. A simple example—if a balancing system relies on user telemetry, that system might poll a store of data, or it might wait for event triggers based on the data. The data might be persisted in a database, or stored in runtime memory and discarded at the end of the gameplay system. The telemetry might be aggregated across multiple user play sessions or might be confined to a single instance. The point is—all of these software development practices are appropriate depending on the use case that they are applied. As such, it is more important to have a conceptual understanding of the balancing loop that can be agnostic to a game's engineering context.

These diagrams also assume a working knowledge of common approaches in AI development—I will not deep dive into implementation strategies of an evolutionary algorithm, for example. That being said, I will describe the use case for each method and provide publications that demonstrate the strategy for further reading. These diagrams also can be nested in different ways to get more complex architectures; you might slot in runtime adjustment within the economic modeling diagram to get dynamic balancing for game economies. Going further, you could slot reinforcement learning into

dynamic balancing, demonstrating that your economic balancing strategy will operate at runtime with an online reinforcement learning approach. Doing so gives you hybrid architectures for game balancing with a foundation for the necessary components of your desired system.

Finally, I try to use a common language across these diagrams that carries meanings across different methods, enabling translation between models. A key note—for both players and designers, the symbols are listed as "designer-agent(s)" or "player-agent(s)". This directly suggests that when we are considering these methods in and out of procedural methods, we can automate certain elements of the method by choosing whether a human or automated agent occupies that step of the diagram.

Design/Playtesting Loops

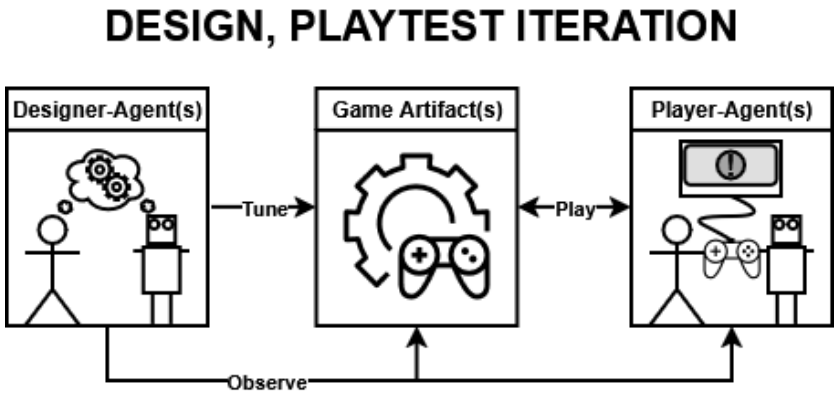

Figure 6.1: The classic iterative process of playtesting. In this example, a designer-agent releases a game artifact, which is then played by a player-agent. This interaction is observed by the designer-agent, which enables further tuning. This pattern forms the foundation for all balancing strategies, procedural or manual.

The cycle of building a playable version of a game, releasing it to playtesters, observing, tweaking, and doing it all over again is a tried-and-true method of game

design that works well in the context of game balancing. Common game design frameworks highlight the importance of seeing the players' get their hands on the artifact, seeing the dynamics of their game in action while observing the reactions of players over time [137, 420, 427, 549]. The observations from this playtest can be qualitative or quantitative, both providing value to the designer in different ways. During interviews, Brie Chin-Deyerle mentions the importance of watching gameplay videos from their playtesters; Oscar Gonzales spoke about how even basic game metrics from playtesting with real players allowed him to understand where different game elements had too much or too little usage.

The simple diagram above should be seen as a container for most other balancing strategies. Highlights the importance of the balancing tuner (designer-agent), the game artifact itself, and being engaged with by a player (player-agent). It demonstrates that balance is borne out as an emergent trait between the player and artifact, and the designer must use some form of quantitative or qualitative observation to understand if their method was a success or not. Importantly, the presence of a designer and a player tells us something important about the proceduralization of balance. When we attempt to automate sections of game balancing, we need to conceptualize an agent that acts as a game designer, and an agent that acts as a player of said game. Replacing one or both of these components allows for automation or novel balance patterns to emerge while retaining a focus on the player (even if they are translated into an automated agent).

Although generic to most game design, the prevalence of interviewees and papers mentioning the power of this method indicates that it plays a central and inspirational role in the game balancing process. To place it in context of balance (according

to our definitions in chapter 4), we should view observations collected through the lens of player fairness. As you will see in the next method, the tuning of the game artifact is greatly benefited by conceptualizing the game configuration through parameterization.

Parameter Tuning

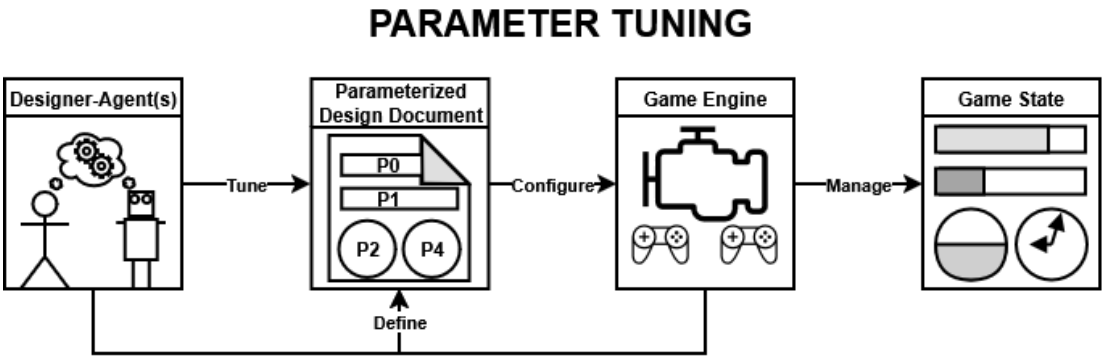

Figure 6.2: Expanding out the design-playtest loop, parameter tuning means extracting key elements of the game into a design document. When this document is modified, the behavior of the game changes. Parameters could be tied to any rule or game element: a character stat, an enemy behavior, an audiovisual element. What's important is that the design document provides an interface to quickly impact how the game operates at runtime.

Games can be interpreted as data-driven software. Regardless of the quality of the code, the games carry rules and entities that are defined at design time with their outcomes reflected at runtime. While it is best if these values are clearly separated from logic, even hard-coded behaviors and properties count as parameters to be tuned. The importance of parameters in the construction of games is most obvious in the theory and practice of automatic game design (AGD). In AGD, an entire game artifact is interpreted as a collection of data that must be configured before being sent to a runtime environment for operation [31, 506]. A specific example demonstrating this is in "Automatic Design of Balanced Board Games" [189], where a general game playing

platform (Zillions of Games) is used to generate games through rule/property definitions via parameters such as board and piece types. The generated games are then searched for balanced examples. Another demonstration of the impact of parameterization lies in projects such as Video Game Description Languages (VGDL), looking to encode these rules into high-level descriptions that can be edited by a user to describe a game in a software-interpretable way [115, 419].

We do not need such a high-level or experimental understanding of definition to understand parameter tuning in the context of game balancing. It is as simple as looking at a set of game behaviors or concepts, understanding how system definitions (numbers, enumerations, etc.) influence their outcomes, and attempting to tune those system-defined variables to produce the best gameplay. If Mario's jump feels badly tuned, a designer can look at a variety of parameters to change—the warm-up, cool-down, upward acceleration, falling acceleration, and so on. While the space of permutations of these elements are overwhelming, a combination of designer intuition and the methods below make grappling with it possible. Chapter 8 covers a technical approach that I use that emphasizes the importance of a parameter list in the game generation and tuning process.

You will see parameterization as a key part of most balancing methods—to impact balance, we need levers and tools to change the system. Parameters give us those handles and knobs that we need to get the job done.

Analytics

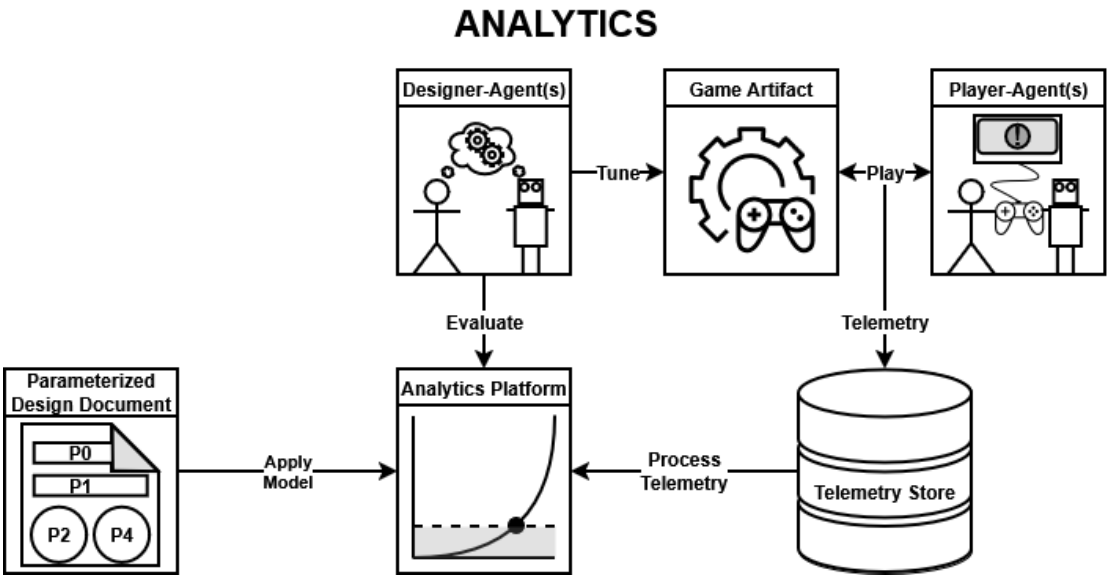

Figure 6.3: Turning to the "observe" portion of the game-design iteration loop, analytics give us a measuring stick to understand what is going on in play. Notably, analytics can be pushed to visualization tools to help a designer manually tune a game, but the same methods can be encapsulated within a game artifact to enable dynamic balancing. Analytics also forms the basis of balancing driven by "mathematical models", wherein a static set of game parameters are measured against one another.

If a designer is using parameters to tune their system, we can view every tuning that a player engages with is a condition of an experiment. With a tuning input, what kind of behaviors and observations do we get from the player as an output? More importantly, how do we make those outputs interpretable so that we can make well-reasoned design and tuning decisions from them?

This is where analytics techniques come in. Playtraces [436], heatmaps [528, 109], win/loss statistics [186, 245], distribution of playstyles [182] all can help a designer understand where the balance of a game is faltering. Analytics can be used from static or dynamic data sources. Static contexts mean looking at the parameterized game and looking for mathematical models that can make all of its elements come together in a satisfactory way. A major proponent of this is Ian Schreiber, whose book "Game Balance" is oriented around the usage of such models to balance spreadsheets of game parameters in search of a satisfactory configuration [422]. An example given by Ian revolves around the costing of different trading card units across a larger set by investigating the distribution of different relevant parameters of the cards.

On the other end, dynamic relies on telemetry from gameplay or simulation. This is often taken from game state and end-of-game outcomes—how did the player-agent impact the game over time, and what were the results? When this telemetry is aggregated, analytics and visualization techniques can be used for designers to surface the when, what, and how of a game's balance, identifying the culprits and times when the game's scales fall out of equilibrium. Johannes Pfau's work focuses on such analytics in the context of MMORPG's, using approaches to understand varying definitions of balance through processing of class usage data from a large body of users [369, 370, 371].

Analytics also play a role in more complex runtime balancing systems that aim to adjust balance during play. By interpreting analytics from a game state, the game itself can make decisions about difficulty and fairness without a designer manually tuning elements of the game. This concept forms the foundation of evaluation functions in more complex methods I detail below.

Game System Interactions

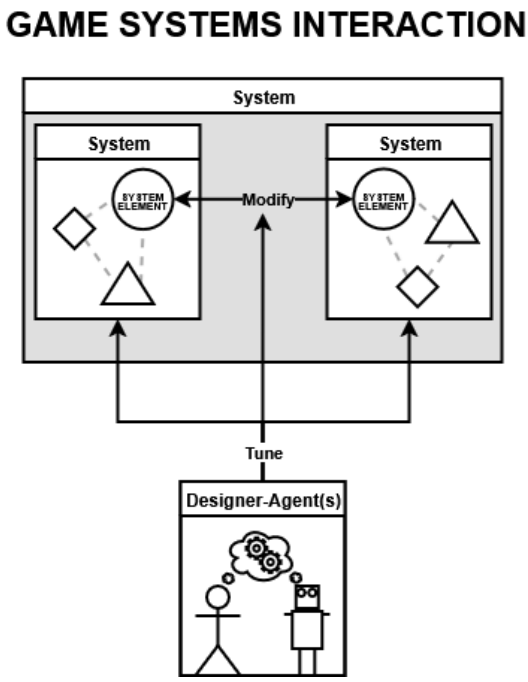

Figure 6.4: Moving towards the game artifact itself, games are composed of relationships between entities that we can view as "systems". These systems seldom live in isolation—different components impact other systems, or the overall behavior of the system impacts other systems. In this conceptualization of balancing, one needs to not zero-in on a single system, but observe the downstream impacts on other systems as well.

While it would be nice for games to have completely isolated, simple systems for us to tune, it is almost never the case that a game system defined in one context is wholly inert to other game systems. A game economy does not exist in isolation—it is tied to the combat needed to engage in the economy, the unlocked items available based on player progression, and so on. In balance tuning, we must hold the same to be true. Each tuning will have knock-on effects with other systems. During our interview, Melody Ju put it thus: "...all the features and systems are working together to [designer] intent..."

System interaction means that sets of systems and interacting game elements might cause unexpected outcomes in one another. Making an object expensive could make it harder for a player to acquire, but it might also encourage arbitrage somewhere else. Mike Sellers' "Advanced Game Design" discusses game parts and systems, noting that parts are components of systems that have a progression over time in accordance with the system's operation [427]. These systems form a network of overlapping and interconnected systems that eventually make up the game as a whole. In this context, tuning one system might require a designer to collect observations and data about not only the system of their focus but also the networked systems and the interaction mechanisms between them. This fact means that when we are considering how we evaluate the balance of a game or consider optimal constraints for our balancing architecture, we must often make our observation lens apply to a larger scope of gameplay and state outside of the specific target or system we are trying to manipulate directly.

Logic Programming

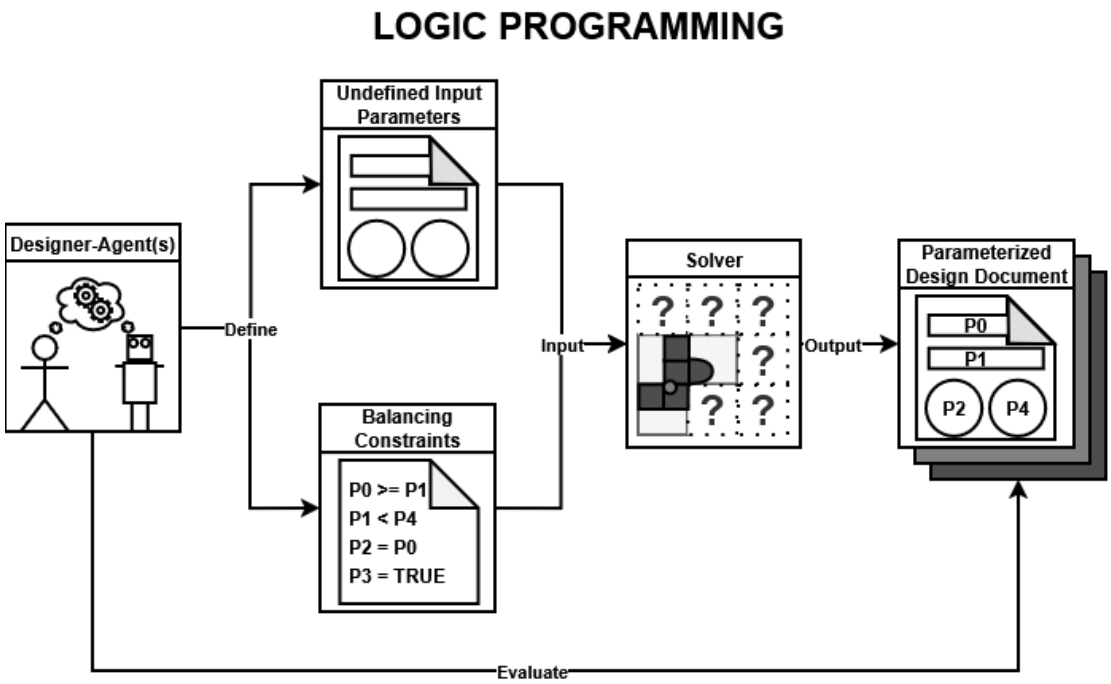

Figure 6.5: Logic programming offers an inverted mode of discovering balance. Instead of building solutions and testing them, you instead take your parameter document and describe valid relationships between properties on that document. From there, a solver will return a set of documents that can be used in the pattern shown on parameter tuning.

Logic programming is an approach in game balancing that is understudied; it falls nicely around the idea of setting goals and parameterizing input and looking for a system to find the optimal solution for you. In this architecture, a blank parameterization of relevant game balance attributes becomes an input. The designer then must describe valid and invalid relationships between those parameters—enemy health must always be in range of a potential combat skill, a game's timer must be proportional to the minimum number of moves required, and so on. In the context of logic programming, these relationships (constraints) and input parameters can be fed into a solver, providing a set of possible configurations of balance parameterization that can then be evaluated or used in practice. Smith et al. [449] describes the value of using this approach in his BIPED system, where the output of potential generated games from a solver can in turn be put through automated playtesting to judge the quality of the solutions.

While underexplored in context of game balancing, the ability to conceptualize the relationships between game parameters and their downstream outputs on game fairness is interesting both in the logic programming outcome and also as a design exercise. It challenges a designer to concretize the construction of their game's balance, understanding how constraints of balance targets and their relationships lead to positive or negative play patterns in regard to fairness. This is explicitly shown in Smith et al.'s *Ludocore*, where the player model and context are shown as interfaces to create the potential for agent interactions while considering aspects of player persona and environmental context [450]. As such, it might be the case that reframing a balance parameter sheet in terms of possible constraints alone is a step towards balancing a

game, even without a solver present.

Runtime Adjustment

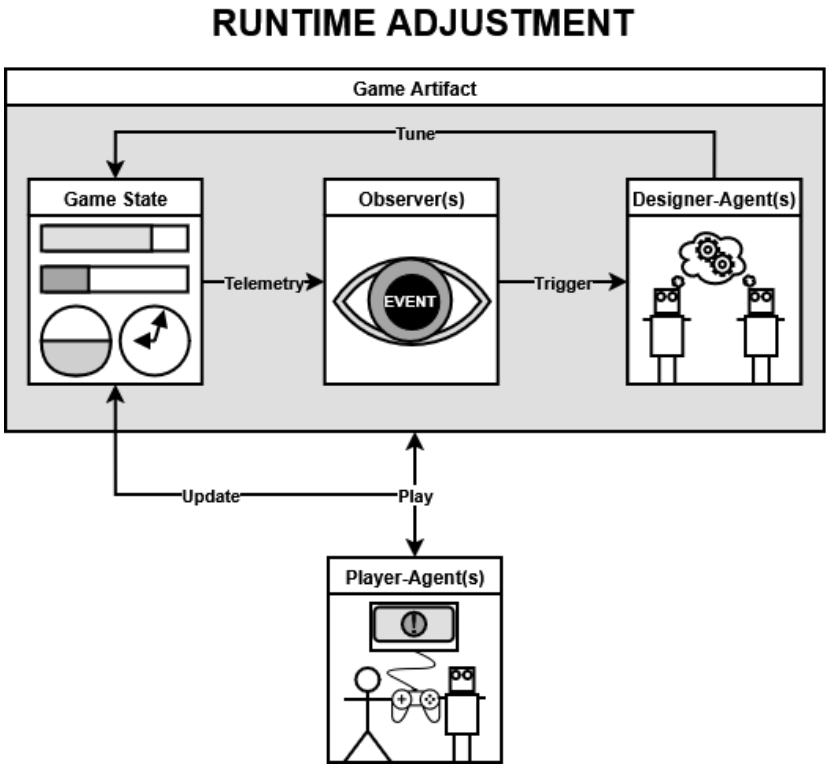

Figure 6.6: Runtime adjustment takes other methods of balancing and places them within an automated context. What's important about runtime adjustment is the presence of a game state (the interpreted and updated parameter sheet). Based on this state, observers (e.g. event-driven or polling architectures) can deliver notifications to a designer-agent, which then in turn decides to update the game state.

Runtime adjustment forms the basis for any form of dynamic game balancing strategy (and, by extension, dynamic difficulty adjustment). The core of runtime adjustment consists of just three essential parts: a game state that can report on what is happening in the game, an observer that either pushes/pulls game state information and orchestrate it downstream to some form of agent, which in turn modifies the game state accordingly.

This pattern allows for flexibility in the usage and implementation of any runtime adjustment. The designer-agent could be a simple heuristic, a sophisticated reinforcement learning agent, or something else entirely. This designer-agent must be automated as part of the game's runtime, as it must operate in environments where a human designer is unavailable (or incapable of matching the speed of gameplay). The observer can be any sort of system that evaluates the game state for meaningful changes, although event-driven architectures generally are preferable for many game engines. The game state is (again) a parameterized representation of the game that exposes elements of the game design to be manipulated and reported on by the overall system.

Many forms of game balancing meet the runtime adjustment criteria. On the simpler side, racing games like "Mario Kart" employ tactics like rubber-banding in order to take players with poor placements and reward them with high-powered items or speed boosts [157]. On the complex side, AI directors and experience managers as a whole use computed measures based on player models of emotion and drama in order to tune the balance and intensity of a game [438, 252]. For a deep dive into the technical side of such directors, see Chapter 10. To implement runtime adjustments (a form of procedural balancing), the game designer must act as a meta-designer, hypothesizing

what they would do moment to-moment to tune for an imagined player and turning that into a procedural system.

Information Visibility

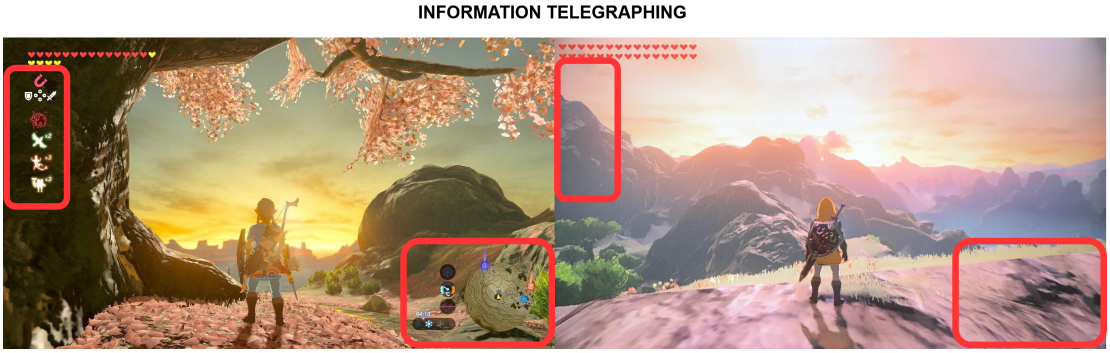

Figure 6.7: While somewhat more difficult to conceptualize as an architecture, the information telegraphing involves two modes. The first, information availability is parameterized and toggled to prevent/enable the player from learning something about the game. In the above example, a player of *Breath of the Wild* can turn off their heads-up display, making them rely on environmental cues to decide information. In the second, meta-information about the game becomes available to a community, influencing how they play the game.

This is the one form of game balancing that resists being put in a conceptual model, as it directly relates to aesthetic properties of game balancing. To still give a concrete example of such an approach, I have instead presented a pair of screenshots from "The Legend of Zelda: Breath of the Wild". On the left, the head-up display shows a map with points of interest, weather information, and skill availability. This information gives the player quick access to key information about gameplay without much effort. On the right side, none of these elements are present. To determine the weather, the player must observe the environment. To navigate to the next point of interest, the player must explore or maintain an internal knowledge of the map layout. Without skill availability, the player must keep track of when they last used an ability before using one again.

Such a simple change of how the player receives information can radically change the challenge and feelings of fairness for a player. To accomplish this, designers

still need to parameterize game elements that convey information (in our example, heads-up display icons) and tie them to controls. The designer also needs to have a strong aesthetic knowledge to understand the down-stream impact of how such aesthetic changes will change the balance of the game overall.

Information telegraphing does not only occur in the context of the game environment. The knowledge a community develops through conversation with each other and the designer also impacts how balance is tuned. For example, the game "Tunic" deliberately hides information in its instruction manual, but players sharing knowledge about these obfuscated instructions can make their interpretation much easier [71, 337]. Like spoiling a movie, game guides and strategy threads can change how balance is interpreted by an audience. For example, a community with deep knowledge of an MMORPG economy might be able to exploit elements of that economy with enough coordinated effort.

Economic Modeling

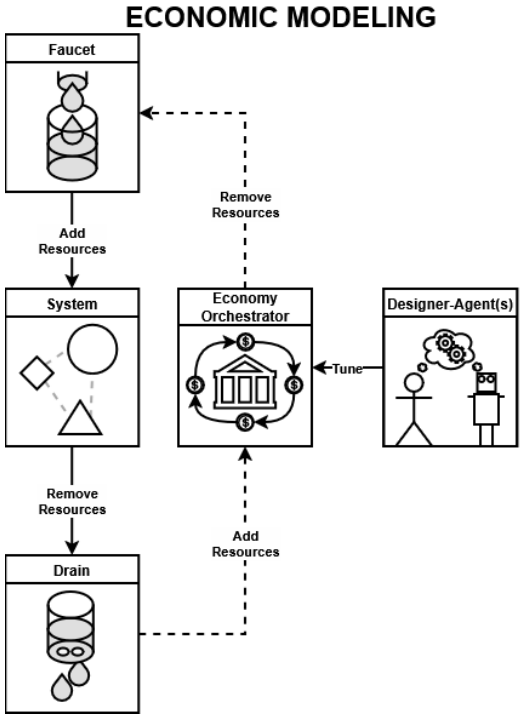

Figure 6.8: This balancing architecture involves the treatment of game systems as economies (implicitly or explicitly). In these models, systems move resources around, interacting with other game systems and elements through faucets and drains, which add or remove resources from a given system. Some form of orchestrator exists in the balancing system, helping negotiate the translation of resources between systems. A designer-agent can tune the systems or the orchestrator to help achieve their desired economic behaviors.

Economies serve as a valuable metaphor for game balancing purposes. As Mike Sellers puts it—"The relationship between the costs and benefits underlying any power or progression curve is an economic one" [427]. Although we may think of balance as fairness, items, skills, levels, etc., dramatically impact how fair a given scenario is. This indicates that the parameterized metrics that drive a given progression can be conceptualized as economic. Such modeling is not just implicit—many games explicitly have markets that directly emulate real-world economies. Going a step further, many games cross the threshold between virtual- and real-world economies. Such interactions between real and virtual economies often drive controversy over access to overpowered gameplay options. Inequality driven by providing players advantages based on their ability to actively pay for advantages has been seen by experimentation with Blizzard's *Diablo III* [39] and Valve's *Artifact* [497]. In both games, the ability (or inability) to optimize the gameplay based on your access to real money caused dramatic changes (or outright failure) in these games [256, 478]. This isn't always a bad thing—"EVE Online" is famously enjoyed for the game's in-world economy being deeply tied to real world assets, adding real weight and drama to community exploits and conflicts [166].

On the academic side of things, systems such as GEEvo [407] seek to find balanced economic states through procedural generation via evolutionary algorithms. There is common documentation around the importance of faucets and drains, which move game elements that are conceptualized as economic in and out of game systems. Thus, to perform balancing through an economic lens, one needs to view systems as having this in/out flow of resources, coupled to other systems and some form of economic model that moderates and manages the way these resources are translated into different

resources (currency to items to damage, for example). The player's engagement with these economic models then becomes a challenge of how well they can mentally map the economy while exercising skill to accumulate resources at a reasonable pace.

Player Skill Determination

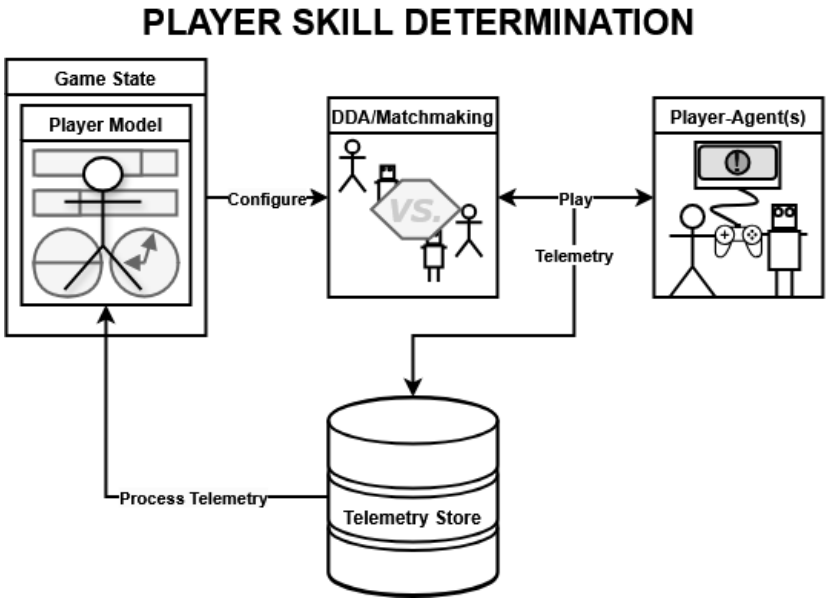

Figure 6.9: If difficulty is deeply connected to balancing, methods of altering the game based on an internal player model becomes necessary. Player skill determination relies on the internal observation of player behavior, aligning it to a player model, and then deciding how that player model impacts difficulty or matchmaking systems. This data is often aggregated through multiple play sessions, building a longitudinal model of the player over time.

Player skill is often tied to the heart of game balancing. The sheer prevalence of work that links the balance to the flow channel is emblematic of this linkage [500, 403, 388, 447, 26, 32, 260]. As such, balancing strategies need to have some manner of observation in order to guess the skill levels of players in order to get a proper tuning of the game overall. This tuning can be done through manual means during design, but is often written about in terms of runtime adjustments to difficulty.

Notably, this area of balancing architecture includes all of the dynamic difficulty adjustment and matchmaking procedures. In both cases, the system does not operate unless there is some understanding of how skilled a player is. In multiplayer environments, this determination takes place in the form of systems like ELO, where rankings help place players of similar skill against each other [118]. Andrade et al. [13] did work investigating the training of agents so that they can meet the needs of different levels of skill in players, showing how NPC behavior and overall game dynamics can be tuned to meet player needs in single-player contexts. In their experiments, Andrade demonstrates how difficulty can be tuned for skill both at design time (the difficulty of the agents) and run time (how and when agents are assigned and what behaviors they use).

The architecture can be abstracted to a relatively simple pattern. A game state contains some model of the player, which can be evaluated to configure a system that is tuned for difficulty or match making. The player participates in the tuned output, producing telemetry that can be used to update the player model for future tunings of gameplay.

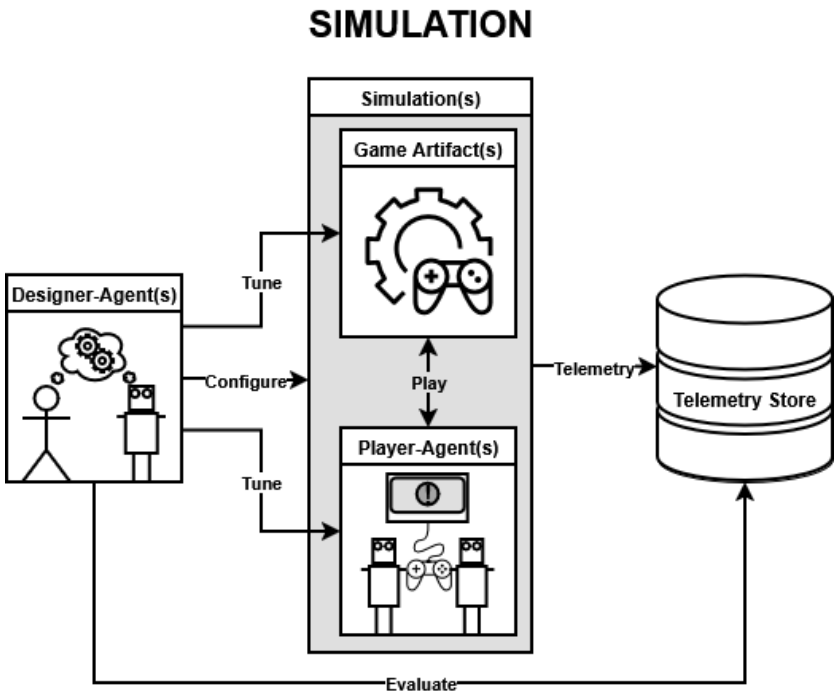

Figure 6.10: Moving closer to full balance proceduralization, simulation gives a designer-agent the opportunity to see repeated runs of a game without human playtesting. If optimized well, simulation can generate enormous corpora of data on which to base future tuning strategies. Multiple variations of the game can be simulated as well, producing a landscape of potential parameterization options to evaluate when balancing.

Simulation

I have noted throughout this document that balancing is expensive—especially when it comes to collecting large amounts of data from human playtesters and tuning accordingly. Simulation in the context of game balancing aims to help solve this problem. In short, simulation runs instances of games with automated agents, recording metrics for analysis and future tuning by designer-agents.

Simulations consist of a harness that determines the number of instances, starting configurations/seeds of games, etc., that need to be observed. The harness is responsible for spinning up game artifacts and the player-agents who are involved in the

simulation, as well as the management of telemetry reporting for all simulated instances of the system. This telemetry store can then be slotted into any number of other balancing architectures—analysis, skill determination, economy, and so on. Simulations are a potent component in procedural balancing because they allow for tuning in high volume before transitioning to human playtesting.

There are some weaknesses in simulations that prevent them from completely replacing the balancing of human playtesting. The ability for player-agents to match the behaviors and progression of human players is an infamously difficult problem. Players talk with each other, develop skills, use unexpected strategies, and so on. While "good enough" agents can get us fairly far, they cannot take us the distance when trying to evaluate real players. There is also a fair amount of technical complexity in standing up a simulation—the game must be performant and scalable enough to run at reasonable speed for the simulation, and custom agents are inevitably needed to make the simulation work as intended.

All of that said, simulations are useful to quickly get landscapes of a game's balance and essential to many automated balancing patterns (especially search-based and reinforcement, covered below). Simulations are used frequently as evaluation methods—above mentions of *Ludocore*, *GEEvo*, among others all use some form of automated playtesting in order to simulate and evaluate game outcomes [407, 450]. I leverage simulation approaches in all the technical work I present in this dissertation, presented in Chapters 8, 9, 10.

Search-Based

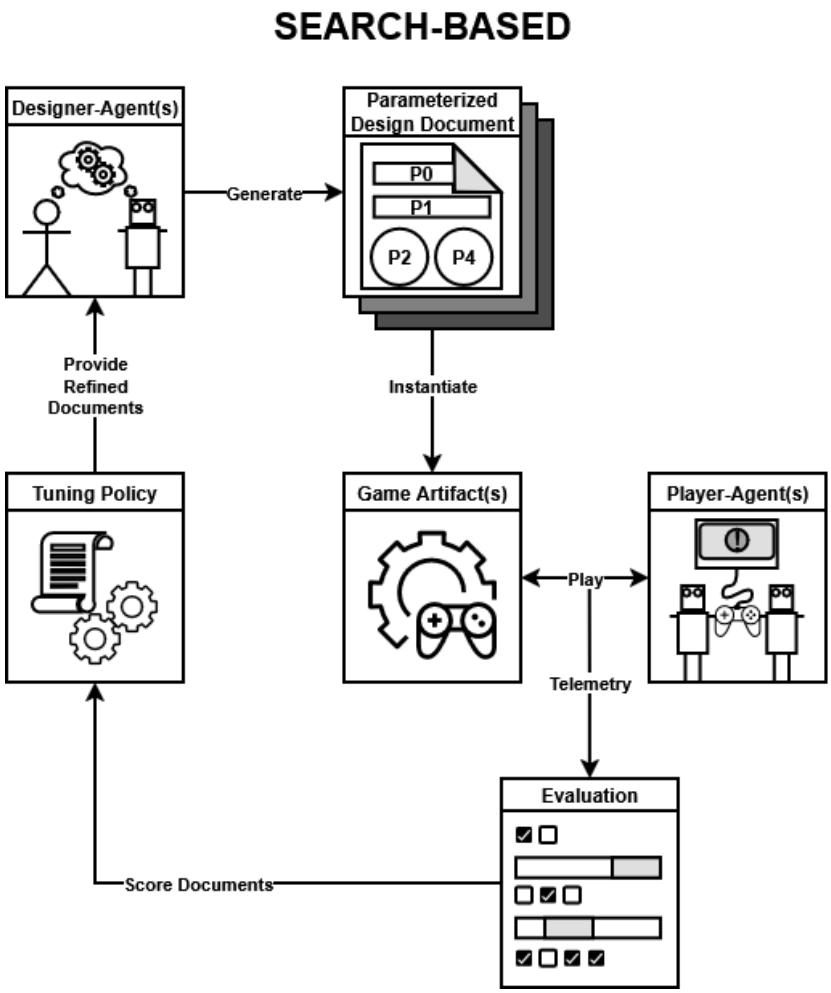

Figure 6.11: Search-based balancing strategies consider a corpora of game artifacts and their runtime metadata and attempt to find ideal compositions for game balance. Search strategies necessarily have some form of evaluation or selection heuristic, which helps the loop identify which examples are optimal. That being said, what evaluation/tuning policy is selected and tied to a particular search strategy is up to the implementer of this architecture. Evaluations, however, should be driven by the player-centric perspective of a successfully balanced game.

Search-based strategies use a catalog of possible tuning options, attempting to find the best configurations of parameters to satisfy some evaluation criteria for the designer. The corpus of parameter documents can be written by-hand, but this approach is often coupled with a generator tied to the parameter document. This approach allows for the exploration of a large balancing space with low designer intervention, as well as the ability to churn out unexpected or edge cases that are interesting or point to system failures.

In search strategies, three components form the core: the parameterized design document, the evaluation of gameplay, and the tuning policy. Search approaches take many forms; the literature has used evolutionary [323], tree-based [98], heuristic [506], to give a few categories. What is important for all of these approaches is that there is some set of game artifacts with variation. When these are played (either by human or automated agents), an evaluation is performed to "score" how well a given parameterization fits a designer's definition of balance. This evaluation is used in the tuning step (where the bulk of the search algorithm lives), either providing an ideal document to a designer-agent or indications on how to iterate the next generation of parameterized design documents.

When automated, search strategies inherit the limitations of simulation-driven strategies, but also provide the advantage of being capable of self-tuning and handling a large amount of tunable parameters at once. Technical experimentation concerning this style of procedural balancing is presented in chapter 8.

Reinforcement Learning

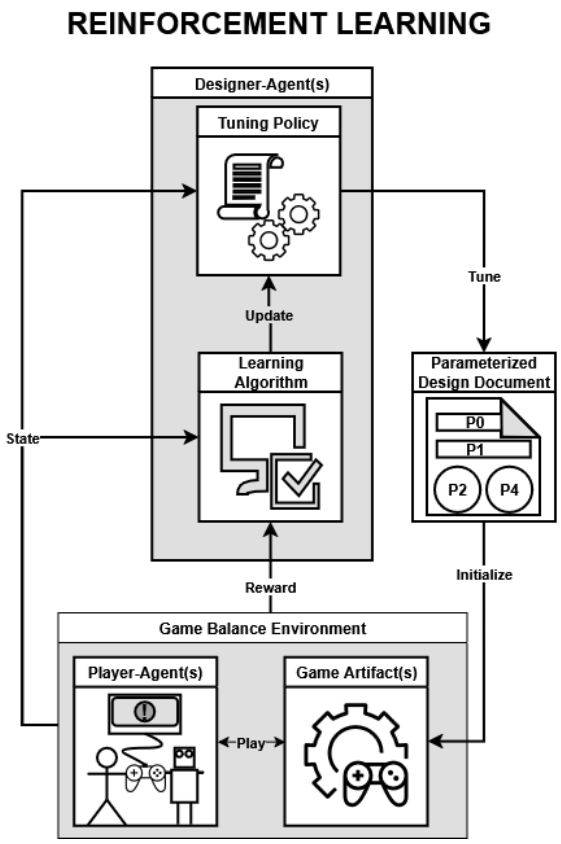

Figure 6.12: Reinforcement learning uses a learning algorithm to update a tuning policy, which in turn alters elements of the game state to meet balancing goals. You can conceptualize the learning/policy combination as a representation of a designer who is tuning the game with each loop through the system. RL can be used with both player-in-the-loop and simulated environments, but suffers from controllability and a tendency to over-optimize.

Reinforcement learning has been the focus of many recent studies on game balancing [342, 388, 15, 291, 304], likely because it reflects an opportunity for behavior design to follow a somewhat more dynamic approach to handling large amounts of parameters and potential downstream impacts. While reinforcement learning is often traditionally viewed in concepts of specific agents within a game environment, viewing the subject in the concept of balancing means that the algorithm is performing actions to tune the game artifact, not necessarily a specific NPC in the game itself (unless that NPC is key to the game's balance). As such, the designer-agent consists of a tuning policy that acts to alter a design document, which in turn generates a game balancing environment (likely a simulation, though it can be a collection of player gameplay sessions as well). A reward guides the learning algorithm, which, in conjunction with state updates, tells the tuning policy what future balancing adjustments are needed.

What the parameterized document covers can be narrow or broad, though most studies tend to restrict it to a smaller domain (e.g. how skilled an NPC is against a player) [14]. That being said, the problem of handling a large combinatoric space of parameter tuning means that RL approaches might be well suited to larger design-tuning problems. There are some weaknesses in using RL (and, to an extent, search) algorithms in balancing. That is, prioritizing pure optimization of a game environment often leads to narrow and uninteresting adjustments [408]. Although this can be addressed by being clever with rewards, design documents, etc., the cost of such fiddling and training might negate any benefits that reinforcement learning might provide in a balancing scenario.

6.3 Converting the Balance Framework into Procedural Balancing Systems

This chapter covered the methods and goals of game balancing, and the previous two chapters covered how we should define game balancing and how we can implement it from a framework perspective. That being said, the point of my work goes beyond solidifying the theory around game balance. The theory I present covers the "player-centric" aspects of this document—it helps us understand who we are balancing for and what we can do to balance. The next steps we take are to translate how this framing of balance can now be properly proceduralized, achieving the 5-10% gain in efficiency or quality that I dangled in the beginning of this work.

The way in which the framework, targets, and methods have been presented suggests to us by themselves a way to go from balancing goals to automated systems to help achieve them. The usage of the terms "designer-agent" and "player-agent" hinted heavily at the approach we will be taking. Namely, our balancing automation serves to leverage AI patterns where the agent tinkering with the system takes the role of the traditional designer, with the implementer (me or you) becoming a "meta-designer" that manages how the automated designer does their work. The "player-agents" test the system, telling us how things went and emulating the folks that we would like to enjoy our system and meet our standards of fairness down the line. Nesting and combining the above architectures starts to produce systems capable of sophisticated dynamic balancing (such as in FighterDDA) or novel AGD and PCG patterns (such as in BrawlerAGD). They also tell us the ways that a designer might need to aggregate

simulation and automated balancing data (such as in Playtrace Arc Search).

With our methods in mind, I will close this chapter out by detailing how to move from our framework to procedural implementations of balance, which I will then demonstrate through implementation in Part III of this document.

6.3.1 Audience and Goals as Objectives and Fitness

When discussing a procedural system for balancing, we want a way to understand the quality of an output balancing configuration. This takes many forms and can be seen as a scoring against what a designer wants to see emerge from a game artifact. I demonstrated in chapter 3 that this is a tough transition when balance is an emergent phenomenological manifestation of fairness. The designer's primary task is an exercise in translating their ideas about what elements of fairness in gameplay are translated into what metrics the game state is capable of producing.

This process requires a cycle of hypothesizing, implementing, evaluating, and repeating. However, our framework provides a guide for how we can more accurately and efficiently target our evaluations so that we are measuring what we aim to measure. Namely, knowing what sense of fairness we want from gameplay and from whom we want it helps us derive the shape of a fitness function that will properly measure the balance we want to see in a game. If we know we want players with a long experience in a genre to experience stress and release, we can look at the usage and availability of their resources, their play time, their time and location of failures, and so on in combination to understand if we are applying the correct amount of pressure at any given point.

Although these metrics are often consolidated down to singular goals, consid-

ering them from different dimensions and setting multiple objectives to be achieved is also a reasonable goal for game balancing. If we want to see equal win rates between players, but also want to see a wide range of play options considered, we might seek to make both of these evaluations also balance with one another, only accepting solutions where both are cared for appropriately.

The mapping of real player preferences around fairness and the computational metrics used to evaluate it in systems is a challenging question that cannot be answered without some amount of user research and/or playtesting. Without knowing your players, preferences, and end-goals, any definition of quantitative measurement of balance in game state becomes a shot in the dark driven by intuition. While designer intuition is very powerful, the first two steps of the framework when considered in terms of procedural balancing suggests concerted effort needs to be placed on identifying what feelings of fairness someone should experience during gameplay, and tying that to valid signals from game state for measurement.

The resulting function from this exercise becomes useful in simulation environments, particularly clear-cut examples such as evolutionary algorithms and their fitness functions, reinforcement learning and their reward procedures, and grading of potential solutions produced in logical programming systems.

6.3.2 Targets and Methods as Intervention and Tuning

If we know what we are looking for and we want a system to get us closer to it, the next step is using the targets and methods presented in this catalog to get to our end-state. The targets tell us what we can and should tune in any given game

environment; the methods tell us how to tune them.

As such, we should look at targets from the lens of parameterization. If targets represent tunable items that impact a game's balance and we need to automatically manage them, then targets need to be changed to a state that can be easily changed. A parameterized design document is pulled from this process—it is a set of configurations driving engine behavior that impacts game balance. The actual implementation of this document can be centralized or distributed. All that is important is that the parameters identified and exposed for balancing are taken into account during the automated tuning lifecycle.

Methods then represent approaches to manipulating, mutating and modifying these documents for future game engine configurations. In the methods section, I discussed analytics, player skill determination, and runtime adjustment. We can piece these together to manage a parameter document—analytics drives insights used to determine skill of players through a model, which then adjusts (an instance of) the parameter sheet at runtime. The way in which the manipulation of the sheet occurs is up to the designer, and different methods have advantages and drawbacks. An evolutionary approach might work, but could take a long time to run and converge at local maxima. A heuristic might be ideal, but take a long time to design and fiddle to get right. The point is that the goal of player-centric procedural balancing is to get this parameterized document to the ideal tuning at the ideal time to evoke the desired sense of fairness in a player.

6.3.3 Evaluation as Designer Observability

Our goals have set up our fitness function, our methods our method of generation and modification—our evaluation step pulls it all together into a loop that is observable to a "meta-designer". A game state has key traits we use to tune balance pulled into a parameter sheet, we mutate that sheet, and apply some sort of fitness/reward/value function to understand the success of our tuning efforts. However, this internal evaluation is not quite the same as the designer's evaluation of the quality of the system. While they *can* be the same thing (the designer takes the fitness score from an evolutionary algorithm as the main indicator), a designer might need different criteria for when/how to end the loop of a procedural balancing process. This exit criterion could be convergence of output, all gameplay metrics falling in a given range, or a certain pattern of behavior observed during gameplay.

This step is where we exit the procedural part and necessarily drop the game tuning back into the hands of the designer. The designer, now with a procedurally tuned artifact, needs to put it in the hands of their audience. If the tuning was successful, the remaining tuning loops of the artifact should be decreased, or the overall quality and insights about the game should shine through in future designs. In the event of failure, the designer has a few options to tune the tuning procedure. They could investigate the signals they are using for game success. The modification strategy of targets could be reconsidered. The targets that are relevant to game balancing could also be taken into account. In this scenario, the heuristics provided by the game balance taxonomy and framework provided in previous chapters come back into play.

6.3.4 Tying it Together

In my formulation of player-centric, procedural balancing, the act of automated balancing is a conversation, a meta-design loop between a designer, players, and tuning system. Emphasizing a theme throughout this work, a procedural process will not entirely replace a design/playtest iteration, but it will augment and optimize such a process. A designer can target a definition from the taxonomy, form a heuristic from the framework, translate it into a procedural pattern, and then evaluate it against players. The designer can iterate this process until they are satisfied with the playtests, abandoning the procedural tuning when finished with the task. This framework also enables the construction of runtime experience managers focused on fairness, enabling new dynamics and affordances in games that center on making a game environment more or less fair based on gameplay. There is drama in feeling outmatched, trying your hardest, coming out on top—or the opposite, ending up feeling paranoid and scared for what is sneaking around the corner. Procedural balance is thus not only a tool to achieve a passive perception on if a game is fair—it becomes a design strategy to tell stories through how we feel fairness in a system over time and interaction.

Chapter 7

Evaluating Balance

"There's no right way to balance. There's a way you balance in order to trigger a certain effect in the player that then manifests. Even in the body."

Martin Pichlmair, Co-Founder of NEON AURELIUS and Broken Rules studio

A core of my work is the acknowledgment that we cannot outright automate away balance. We can optimize, we can learn the steps to proceed, but we ultimately need a human to evaluate. So what do I do when I am trying to proceduralize or automate subsections of the balancing process? I realize and fully admit there is a tense tight-rope that I must tiptoe through. When I try to automate balance tuning, I need to apply as many qualitative tools as I have in my arsenal to translate human judgments of fairness into encodings my system can read. This is where evaluation comes in, and

that is the subject of this chapter.

Throughout the rest of this chapter, we will cover the main places where evaluation becomes useful, from both quantitative and qualitative perspectives, with a focus on quantitative. While we introduce qualitative methods, the research and shared design work on playtesting is well developed and is likely better addressed from those sources.

7.1 Playtesting: Returning to Emergence and Qualia

I established early on that understanding balance ultimately lies in the hands of the player. This is a critical insight when considering procedural balance methods—we can *never* reach a balanced game purely through procedural or quantitative approaches. We need to know the responses of the player through their lens, which can only occur if they play the dang game. As such, we should never exclude this step when trying to evaluate whether a game is balanced. Not that I claim to, but expecting a game to be balanced through any means without a player telling you their opinion is an unrealistic mindset.

We have touched on established methods for collecting qualitative insights during playtesting. "Think aloud" protocols can tell you in detail about what a player is reflecting on during gameplay [219, 191]. Post-gameplay questionnaires can give a starting point for where problems are in gameplay—the "games experience questionnaire" is a well-documented example that aims to cover a wide breadth of gameplay examples [264]. Surveys can have a downfall, especially if they are generic. The lack of

specificity around a game can obfuscate where meaningful issues are, and might be difficult to parse without being catered and customized to fit the goals of a given playtest. Replays of gameplay can shed light on player decision making, helping the designer form hypotheses around what might be wrong or missing from the game [292, 211]. Every AA/AAA interviewee and Indie designer interviewed (see appendix A) noted that high volumes of playtesting allowed them to identify where specific problems were in a session, but generally were not great at finding solutions to the problems. Thus, designers might be cautious about trusting player recommendations on how to tune but put more weight on the places where the player expresses frustration or dissatisfaction. Playtesting is a well-documented and important part of any game's holistic evaluation—more in-depth treatment of this topic can be found in the work of Fullerton et al. [138], Schell [420], Koster [246].

Focusing on balancing, qualitative feedback should orbit the taxonomy work presented in chapter 4. Asking players about their global feelings of fairness, their level of frustration throughout gameplay, pacing elements, and competitive viability all shed light on how balanced a game is in regards to a designer's specific goal. Andrade et al. [16], for example, performs a user study focusing on dynamic game balancing that primarily uses Likert scales to find which styles of dynamic balancing players enjoyed the most. While conversions to scales can be useful, getting player's raw feedback also spotlights elements of balance that might not be immediately visible. For example, studies such as Gonçalves et al. [157]'s balancing differential skill levels highlight the notion of believability and the potential for balancing strategies to undermine player's confidence. If balancing is easily detectable to a player, they might get the signal that

"you're pretty bad at this, let me help you out", which is not always a great message for those just trying to have a good time. However, in some cases, where balancing is used effectively as an accessibility tool (as in Gerling et al. [150]), players who would not otherwise have the ability to compete feel increased satisfaction in becoming able to compete with differently-abled peers. To mitigate this potential negative impact, designers often focus on subtle-yet-effective moderators of gameplay, such as the aim-assistance examples given in the previous chapter [29].

Following the concept of believability, it is also important to recognize that player perception of balance overrides any sense of mathematical or quantitative balancing. As Jeff Kaplan put it—"The perception of balance is more important than balance itself' [231]. This element of balancing further emphasizes the potential of aesthetics, community conversation, and available information to be as deeply relevant to how we should consider tuning a game. If the game looks balanced mathematically but still suffers from a player's perspective, one might ask if there are sensory or game feel reasons that are biasing a player's opinion instead of continuing to tweak parameters that will not outright change player perception. Just as subtly changing the balance of a game might drastically change the experience and curve of a game, so too can the aesthetics drastically change the balance of a game.

Qualitative playtesting also plays a role in the building of procedural methods. When we automatically adjust balance or try to quantitatively evaluate it, we must form theories about how our systems reach their balancing goals. However, designers are biased, and players can be unpredictable. Our hypothesized theories of balance may not align with what our players actually value. When considering if our procedural (or

post-hoc, analytics-driven evaluation) methods are working, designers would do well to connect the dots between their models and theories to the qualitative data they might be receiving. Having a dashboard light up green means little if players are telling you the balancing environment is bad. If such a situation occurs, the designers need to return to their systems and tweak it until they see a better alignment with the voices of their audience.

7.2 Analytics

The tendency for balance to orient itself around game-system parameters suggests a linkage to analytics that can be generated and aggregated through gameplay. I have covered common elements of balance that can quickly be evaluated through gameplay data—what are the comparative win rates of players or strategies? At what points does a player quit, die, or take an unexpected amount of time during play? In my interviews with designers, looking at game analytics was referenced over and over again when judging the success of balancing methods (see appendix A for the list of interviewees). Masahiro Sakurai, while also focusing on the feel and holism of balancing a new *Super Smash Bros. Ultimate* character [411], will also point to the numerical representation of win-rates of online play to demonstrate that games are balanced [412]. Seeing a disproportionate usage of a character in *League of Legends* [400] or a unit/faction in *Starcraft* [40] quickly signals to the design team that something is wrong and tuning needs to take place [265]. Analytics form the front-end of the immune system in game balancing—they inform the creators with varying levels of accuracy what, where, and

when something feels bad.

7.2.1 Player Modeling

The utility of player responses and data is in part determined by how well a designer knows who is playing their game and if those players are the folks they are interested in pleasing. If I am developing a Souls-like game, handing the controller to someone who has no interest in video games or fantasy might yield some amusing reactions, but it probably wouldn't tell me anything meaningful about what I need to change the game. While it is admittedly an extreme example, we can extend this to player personas and models writ large—if we are asking the right questions to the wrong players, we undermine our judgments about what is good or bad about our balancing. To handle this task, we can turn to player modeling.

Player modeling refers to the practice of making a system-defined model of how a player might behave, feel, or think before, during and after game [536]. Smith et al. [451] build a taxonomy to help us frame and scope player models, listing domain, purpose, scope, and source as the key components to developing a model of players. Player modeling is key to understanding how a system should respond to the input provided by a given audience, and is a common practice during game design [111] and the construction of procedural systems [362]. Modeling is particularly relevant in adaptive game systems such as dynamic difficulty adjustment (DDA), where modeling perceived difficulty allows a designer to appropriately modify a game's properties to meet a player on their skill level [26]. This is exemplified by Valve's *Left 4 Dead* [498] AI director (AID), which uses a calculated metric for "emotional intensity" as a means to define

gameplay curves. In pursuit of roughly sinusoidal cycles of rising tension, climax, and falling tension, system perception of emotional intensity drives enemy spawn volume and timing [45]. It is important to note that player modeling is often a designed composite of many gameplay metrics. In the case of *Left 4 Dead* [498], there is no raw value measurement of "emotional intensity" transmitted from the player's limbic system; it is calculated from a combination of in-game metrics. Thus, we can easily visualize and operate different dimensions of player modeling through these composite metrics that are created by a game's designer. The ability to understand if your player modeling is 1) correctly tuned to player experience and 2) allows for the correct manipulation of system interaction is, in turn, critical for a designer to meet their intended experiential goals. More discussion on this is detailed in chapter 10.

Modeling can be performed in many ways. With access to a large amount of playtrace data, machine-learning models can be built to identify clusters of players, which can then be used to hypothesize which attributes of those players are the most meaningful [536, 192]. Any amount of data, combined with designer expertise, can also leverage techniques such as the development of player personas to give themselves ideal targets to work with [491, 416, 415, 414]. During my interviews with designers, it was often mentioned that designers would use themselves or close friends as the target player model. Their reasoning was that if they or their close friends found the game fund, it would likely be an indicator that a wider audience of like-minded players would also find their game fun.

Another important aspect of using player modeling in evaluation strategies is that such modeling provides methods that translate a perceived ideal player into a

system-understandable model. This model can then be used to implement game systems or evaluate their effectiveness. For example, player models might be useful in terms of understanding what behaviors players commonly exhibit [536, 276], or what sequence of dialog and narrative beats should be provided in a drama [303]. Models have proven particularly useful for creating specific metrics that represent elements of dramas for players. Jenova Chen of *thatgamecompany* described using an "emotional arc" model to construct the progression of *Journey* [476, 77]. An emotional arc is a designer-defined metric borne of an understanding of what the intended experience of the game is for a desired audience. As such, building a conceptual model (either from designer expertise or specific gameplay analytics) helps designers scope which players are the most important to listen to during iterative development, as well as build computational systems involving evaluation of the player model itself.

7.2.2 Playtrace Inspection

Information from a game state, especially as it changes through player interaction, is a common and well-proven pattern when attempting to understand game balance. Instead of breaking down every potential dimension of potential game state data, I will group them together as *playtrace data*, which frame this collected corpora as the metrics one can collect throughout or as a result of gameplay.

A playtrace from a video game is defined by Osborn et al. [352] as "a sequence of player actions corresponding to one play of the game." As players perform actions in a game, observable metrics are produced as a direct or indirect impact of changing the system. Such metrics might include parameters relating to a game goal (e.g., how

many items have been collected), bug prevention (e.g., has a player reached a checkpoint in a race too early), or updating a player model (e.g., what level of difficulty the player is experiencing right now). Metrics can have many indexes in a recording (every tick) or summarize metrics at the end of a gameplay session (wins/losses). The role of such metrics in design strategies such as Dynamic Game Balancing [16] as well as scoring games for quality in A/B testing [202] helps to underscore the utility of metric production in both design time and runtime. These metrics are frequently trivially available due to their critical role in modifying game systems and are also useful for providing system observability of a game system over a play session. Designers have the responsibility not only to tune input parameters to a game system to ensure a desired play experience, but also to utilize these observable metrics to iterate on their design strategy and confirm their hypotheses about runtime operations of games. The value of producing understandable and meaningful playtraces is, as such, a meaningful step in iterative game design, especially when it comes to human or automated playtesting.

Playtrace collection and analysis of game metrics has been a critical area of game design and research for well over a decade. Drachen and Canossa [110] make the case that gameplay metrics provide critical information about player behavior and that any action a player can take can potentially be measured. Yannakakis and Togelius [535] discuss the importance of linking such metrics with player affect to make available player reactions and sentiment through game play recordings. Wallner [509] discuss the importance of large playtrace and player metric corpora, especially when it comes to the production of visualization tooling such as heatmaps. However, such large sets of playtraces are inherently difficult to parse and nearly impossible to investigate on a

case-by-case basis [286]. This does not mean that investigating an individual playtrace is unimportant—the study of individual games between players has historically been critical to the study of games such as *Chess* [133] and *Go* [393] and is a common attribute of eSports (such as in *Starcraft* [40]) and professional gaming commentary [37, 281]. The ability to understand trends at large in a set of playtrace data and identify critical examples to investigate provides a solid foundation for thorough analysis of playtesting data. We will discuss tooling and approaches to handling such large corpora in Chapter 10. Playtrace data are essential for post-hoc analysis, but can also play a central role in automated dynamic and static balancing systems. Knowing elements of the game state that indicate player performance, pacing, game difficulty, and so on can inform the changes a supervising system needs to make.

7.3 Automated Evaluation

Turning to procedural work, we can combine the data generated by playtesting with analytics techniques to obtain automated evaluation strategies. These practices involve the usage of automated playtesting or simulations to gauge the balance of a game. The techniques can range from simple to astoundingly complex, but the overall pattern is roughly consistent: we have a computational method to emulate gameplay or system construction, and some computational method to evaluate the quality of game balance based on said emulation.

On the simpler, low-cost side of things, creation of spreadsheets detailing system attributes alongside a mathematical model to calculate some form of cost or value

can produce outsized results when finding the distribution of power within a game system. An example might be found in the tuning of trading card units. Each card might have a set of attributes (cost to play, if it draws cards, combat value) that are tied to some abstract representation in a mathematical model. The model itself is essentially an equation, where a net output value is determined by the combination of operations on the attributes involved in the card. The total output from these attributes then becomes a term that can be used to determine the balancing "score" of a given card, which can be used to spot outliers or curves within a global set of cards. Ian Schrieber is a champion of this spreadsheet-oriented approach, noting that it is "The only tool I need to balance something". Indeed, this approach of "costing" game elements and making sure a given set has low computational cost and is easy to understand and implement, making it an excellent first step in getting a view of the balancing landscape of a system. During procedural content generation, such a model can also help designers set bounds on what types of new elements might be valid for generation [422].

In a way, the creation of a mathematical model and balancing of game elements is a rough-grained simulation of gameplay—it approximates the outcomes of gameplay through a low-to-the-ground mathematical model and parametric description of game elements. In approaches such as these, we are not looking to player-driven analytics to understand system balance, but to simulations of play to get a "first-look" at what performance with players would look like. While this form of evaluation is perhaps the most susceptible to misalignment with player preferences or sentiment, its utility stems from the ability to do it quickly and in isolation from a playtesting audience, truncating the number of tweak-play-tweak cycles needed to find an ideal balance.

The other side of complexity from hard-coded mathematical models and cost spreadsheets lies in the use of agents to test game environments for balance. This practice has recently been growing in popularity in both academic and industry contexts, and much of the technical work I present in chapters 8, 9 and 10 is based on the use of such agents. The basic premise is to utilize AI agents to act as players repeatedly, creating synthesized playtrace data that can be used to tune and analyze accordingly. Although not a 1:1 emulation of players, the presence of "good enough" agents means that a designer can get closer to a goal game balance before involving playtesters or increasing the quality of a given tuning session. In industry, studios use these agents in a variety of contexts of balancing—Square Enix used them to identify the difficulty curves of encounters in *SaGa Emerald Beyond* [461] by using reinforcement-learning-driven agents to understand where spikes or unwinnable battles occurred, enabling tuning prior to release to players [319]. Automated playtesting is used to guaranty completeness and score game quality—as seen in *King's* work on *Candy Crush* [185, 242] or Julian Togelius' work on using neuroevolutionary architecture to generate game content [484]. In general, the usage of automated playtesting to evaluate game quality (including balance) is fairly prevalent and has the net benefit of being usable for larger PCG architectures.

The utility of automated playtesting in balance does not end with the evaluation of the game. The bots used for playtesting themselves can be used as *part* of a game's balance, especially as it relates to NPC behavior. Imtiaz and Mujtaba [206] demonstrate this by using RL techniques in creating bots of varying skill levels to cater to players at different levels of experience. In an interview, Dominick Reba noted of

Project M/Project+ AI programming that he wanted to have agents remember player behaviors to change up their behavior and become more unpredictable. When implemented in the running game environment of players, AI agents of varying capabilities produce another variation of balance, one that is not easily understood through cost sheets or static analysis methods alone.

7.3.1 Metagame Analysis

Data from a playtrace is one thing, but the accumulated knowledge and performance of a player base are another; especially when such knowledge is not visible from game data alone. This is where metagame analysis comes in. Metagame reviews give designers hints on the tone of discourse, the information available to players, and the out-of-game tools that players are using to optimize or revise their play strategies. A metagame is understood as the sum of game elements that exist outside of the coded rules of the game [103]. Richard Garfield, creator of *Magic: the Gathering*, loosely defines the term as "...how a game interfaces outside of itself. A particular game, played with the exact same rules, will mean different things to different people, and those differences are the metagame." [171]. Metagames, as such, are composed of commentary between game communities in response to collective or individual play. These communities form dialogs and tools to engage with the game. In *Pokemon's* official competitive format, the dominant teams and strategies change week-to-week (without balancing or rule changes) as players observe tournament responses, attempt to counter common strategies, or develop unexpected team compositions [17]. They also have shared knowledge (not provided in the game) and explicit tooling that helps to build

key game elements. There are exact damage calculators, catch-rate calculators, random number generation manipulation tooling, monster breeding and training resources—the list goes on[1]. The *Pokemon* games themselves make most of this data partially or totally unavailable to players—the development of balance and strategies dependent on these tools and knowledge solely depends on the community and the research they perform into the game itself.

The known information about a game via a conversation between players and designers also informs the metagame balance of games. Perhaps the best example of this is through speedrunning and randomizer communities. In these communities, the discovery and search for glitches are a popular part of the competitive practice of players [183]. This has become prominent enough that automated approaches to finding these glitches (alongside alternate routes) are not only commonly implemented through Tool-Assisted Speedruns, but also researched in academia through automated playtesting strategies [91]. Decompilation and data mining projects carried out by game fans give communities even greater knowledge of how games work, also enabling new strategies [495, 333]. Streamers and Youtube commentators gather substantial audiences discussing the way the community gains and puts such metagaming knowledge into practice [315]. The investigation of information about a game and its impact on the balance of said game, in this way, becomes a game and activity in and of itself; one that designers can encourage, ignore, or attempt to shut down.

There is overlap between metagame analysis and analytics approaches. Look-

[1]Damage calculators can be found in `https://pokemonshowdown.com`; catch-rates in `https://dragonflycave.com`; random number manipulation in `https://retailrng.com`; VGC breeding tutorials in `https://vgcguide.com`

ing at the adoption of classes or archetypes in games like *Guild Wars 2* [21] has yielded insights into player perceptions of over- and under-powered builds [369]. Tournament results and card usage rates are frequently used to inform future design tweaks. The presence of an active community during the development of games like *Slay the Spire* [306] was critical to having a balance and tuning strategy at all [537, 153]. Data collection on player sentiment can also be used to judge overall tone of reception to the game or releases; though such scraping of written content from players can wade into murky ethical waters.

Through all of these points, metagaming can be thought of as uncontrolled playtesting, research, experimentation, and development using your game as a platform. Investigating the complex and interwoven components of such an ecosystem can shed light on future balance choices.

7.4 Expressivity and Drama

The above analysis generally revolves around aspects of balance as framed parametrically and through the lens of difficulty or competition. Isolating evaluation to these perspectives scopes out another important aspect of balance—aesthetics in all of their forms, but especially as they relate to expressivity. While balance itself often uses these metrics, the act of balancing is sometimes perceived through the game's sense of drama (in pacing, narrative, game feel) and often has downstream impacts on how the game progresses. I mentioned earlier that simple tweaking of lighting and sound effects can change the player's perception of balance. As such, the way a game's balance

changes over time itself tells a story to the player through feelings of fairness. A horror game will oscillate between feelings of power and helplessness, representing cycles of fear and overcome through its balance characteristics [172]. Power fantasies start the player with little and continuously ramp them up in terms of powerful weapons (big particles, loud sound effects, high damage output, etc.) to help the player feel like they are continuously accumulating strength and ability throughout the game [474]. Tweaking balance is a way to manage player tension over time, and the curves of how this tension is managed tell an implicit emotional story to the player.

The control of tension in games is a well-researched area of game design that uses a paced attribute of a game (e.g. narrative, difficulty) to match some desired curve of the designers. The concept has frequently been used in the context of narrative games, where a (potentially procedural) game narrative meets the requirements of storytelling concepts such as rising action or exposition [397]. In terms of narrative, these examples highlight how dynamic systems seek to have certain properties of tension while enabling diversity in gameplay experiences. Connecting the dots between narrative and systematic tension [446] highlights how these components work together (and sometimes are in conflict with each other) to form a holistic game experience. In this vein, the intensity graph of games such as *Left 4 Dead* [498] mimics the narrative story curves as they are embedded in a game-mechanics and system perspective [45]. Mejerwall [309] makes this explicit in a talk on the implementation of AI directors, where the control of enemy spawns are set on a roughly sinusoidal pattern that is smoothed into statically-authored moments in order to give players a cyclical feeling of tension build-up and release. AI directors highlight how narrative can be driven through dynamic balancing

techniques, which is in part the justification for my research in chapter 10 of *Playtrace Arc Search*. The need to see these curves and evaluate them also necessitates further evaluation strategies, a current gap in the academic literature. As such, chapters 9 and 10 both present techniques to understand the dramatic nature of game elements related to balance by looking at the expressive range and playtrace curvature of games.

Part III

What was *Done* to Test Player-Centric, Procedural Balancing?

Chapter 8

BrawlerAGD (Automatic Game Designer)

"Donkey Kong can actually be quite potent and relevant against the meta in *Super Smash Bros. Melee*"

Arjun "Junebug" Rao, professional *Super Smash Bros. Melee* player [228, 489]

In this chapter (and in all subsequent chapters in this part of the dissertation), I look to demonstrate the definitions and theories covered earlier in actual technical implementations and testing strategies. These represent practical applications of my framework to real artifacts that show automated balancing and analytics strategies in action.

To start us off, *BrawlerAGD*[1] was developed early on in my degree, when I had a burgeoning fascination in AGD, PCG, and game balancing. It blends concepts of AGD and dynamic game balancing, using a coevolutionary approach to try and not only generate levels, fighters, and moves in a fighting game, but also tune them so that they appear balanced in context. The idea was to test balance *and* mechanic emergence to see if interesting edge cases of game balancing could also hold novel interaction patterns (e.g. wavedashing or the unexpected rise of Donkey Kong in the meta of *Super Smash Bros. Melee*).

This work also dives into quantitative and qualitative player evaluations and inspired the "player-centric" elements of this dissertation. When writing this work, I was engrossed in competitive fighting games and naively assumed that all people would view a 50/50 win rate as ideal balance. In user testing, I quickly discovered that this was indeed not the case, even if players found the games engaging. From a system perspective, the "balanced" games generated showed a success, but this mattered little when I hadn't properly scoped out what kinds of player would take place in my evaluation. These realizations showed me that on the one hand, proceduralizing balance was possible. On the other hand, completely doing so is unlikely, and people had greatly different interpretations of what balance is (motivating essentially the entirety of part 2 of this dissertation).

[1]Published at AIIDE 2022

8.1 Introduction

Procedural content generation has solidified as a commonly-used technique in video games, leading to a wide body of research covering many techniques for creating video game content automatically. More recently, researchers have begun to focus on automatic game design (AGD) (e.g. [55, 89]). Here, the generation systems are responsible for more than a single static artifact that will be used as part of a wider game. Instead, they generate interactive parts of the game world, including narrative, game mechanics, and other game elements such as sound effects. Because these systems generate multiple parts of the game, they usually strive to make coherent choices, so that each part makes sense in the context of the other part, a framework called orchestration [279]. In addition to designing entirely new games, this framework can also be used to discover new mechanics that integrate well with the fixed design of an existing game.

A primary challenge of AGD is evaluation: many AGD systems are evaluated by defining an objective function that intuitively corresponds to desirable properties of the game, and show that generated examples score well on this metric [89]. Due to the complexity of games, the objective is often calculated using statistical data about the playtraces performed by an AI. However, even if the objective function is well-calibrated towards desirable gameplay, there is no guarantee that this AI behaves in a human-like way, and thus no way to tell whether the outputs of the system are games that achieve their aesthetic goals.

This is problematic when it comes to balance in multiplayer games. Although there exists literature on automated balancing in multiplayer games, studies often stop

short of performing user studies to determine if proposed methodologies produced artifacts that end-users perceived as balanced. In a study on Real-Time Strategy balancing, for example, human judgments are wisely included in an iterative system that aims to tune a set of parameters to achieve balance. [378] However, evaluations of this system consist of case studies and evaluations using an internal fitness function, leaving the reader with little understanding if a broader or uninformed audience might actually judge produced artifacts as balanced. If we computationally model aesthetically-driven judgments in a game balancing system (this character feels overpowered, this unit is under game tempo, etc.), we should also strive to show in user studies that these aesthetically-driven properties are perceived without priming or bias.

In this chapter, we describe a system that automatically generates games in the brawler genre, and evaluate its performance with a user study (screenshots from gameplay shown in Figure 8.1). We show that the system's evolutionary algorithm successfully controls many facets of actual human play, but with a weaker effect on human perception of those facets. These results from real gameplay provide insight into the circumstances under which automated evaluation is correlated with human evaluation.

8.2 Background

8.2.1 Brawler Games

Brawler games (a.k.a. Platform Fighting Games) are a sub-genre of fighting games where players each control a single character and attack each other with various

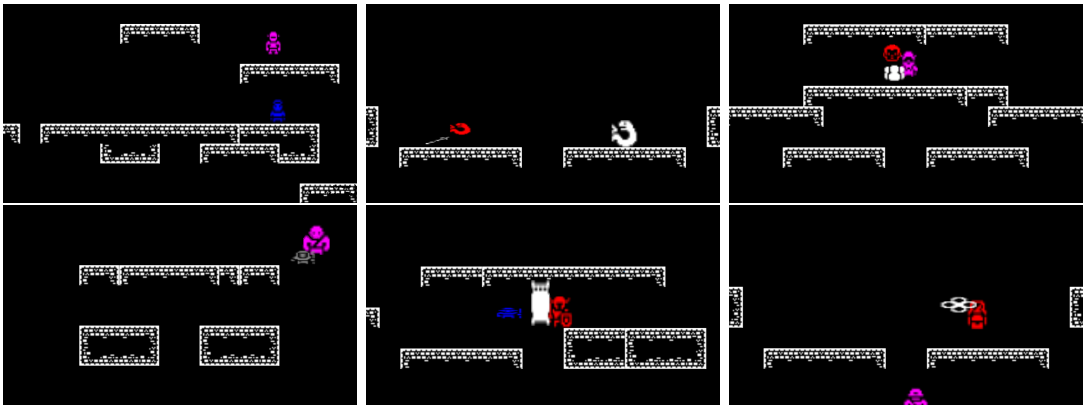

Figure 8.1: Six screenshots from example games. The games' labels from top left to bottom right respectively are: Game A (Evolved), Game B (Random), Game C (Evolved), Game D (Random), Game E (Evolved), Game F (Random).

moves. In contrast to the *Street Fighter* [66] style of game in which players have a health bar, brawler games require players to knock their opponent off the screen. Knockback inflicted to a player is scaled with the total damage that that player has—the higher their damage, the farther they fly when hit. Some examples of brawlers are the popular *Super Smash Bros* games, as well as games inspired by these like *Rivals of Aether* and *Nickelodeon All-Star Brawl* [339, 99, 290]. Brawlers are popular and often host large and competitive eSports communities.

Brawler games are noted for their emergent complexity in their competitive environments despite their ease of learning [85]. Balance and fairness are common concerns for both expert and novice players alike: experts want fairness in tournaments, and novice players want a fighting shot in any given match. Deciding what object properties in a brawling game (damage of a move, layouts of levels, physics characteristics of characters) allow for an interactive, balanced game environment is a complex problem and is the subject of many "balance patches" that publishers apply after initial release.

Our system generates simple brawler games consisting of two characters with one move each and a single level. This simplified game allows us to parameterize the core elements of a brawler game. A complete list of system parameters is given in Table 8.1. Simple changes in parameters have dramatic impacts on emergent game feel. An example: The numerical value of a character's jump force has a large effect on how powerful they are overall. Level design also has a great impact on game balance: levels might have platform spacing that is difficult for one character, but not for the other.

Gameplay differences driven by parameter changes provide opportunities for a parameter-tuning system to produce asymmetrical yet balanced characters. An ideal system should be able to produce games that feel fair with significantly different characters, not only characters that are mirror copies or are so poorly constructed that they cannot functionally interact. We acknowledge that our system does not cover the full space of brawler games (more players, more moves)—the complexity described above scales dramatically with every new mechanic introduced. Due to the paucity of existing work on brawler optimization, this study serves as a starting point for investigating the generation of emergent properties in this specific gaming environment.

8.2.2 Related Work

Table 8.1: A List of parameters used for game generation including ranges for parameter generation. Platform width and location are dependent on the game's bounds.

Entity	**Parameter**	**Description**	**Range**
Character	Ground Acceleration	Acceleration of players on the ground	0-1
	Air Acceleration	Acceleration of players in the air	0-1
	Max Ground Speed	Maximum self-applied speed on the ground	2-10
	Max Air Speed	Maximum self-applied speed in the air	2-10
	Ground Jump Force	Force applied from jump on the ground	1-15
	Air Jump Force	Force applied from jump in the air	1-15
	Mass	Impacts momentum and knockback	0.5-2.5
	Drag	Impacts air control	1-6
	Width	Scales width of player hitbox	0.7-1.5
	Height	Scales height of player hitbox	0.5-2.5
	Gravity	Scales how gravity impacts player	0.3-1.3

Continued on next page...

Table 8.1 (Continued)

Entity	Parameter	Description	Range
	Hitstun	Scales how a character responds to hitstun	0.1-0.3
Move	Distance	Distance between move and character center	0.8-1.5
	Angle	Angle Made between character and move	0-2*PI
	Width	Scales width of move hitbox	0.5-1.5
	Height	Scales height of move hitbox	0.5-1.5
	Warm-Up	Duration of move warm-up	0.1-0.6
	Execution	Duration of move execution	0.1-0.4
	Cool-Down	Duration of move cool-down	0.1-0.4
	Damage	Scalar for damage applied on hit	0-10
	Knockback	Scalar for knockback applied on hit	1-16
	Knockback Direction	Direction knockback applies on hit	(0-1, -1-1)
	Hitstun Duration	Determines base hitstun applied by move on hit	0-1

Continued on next page...

Table 8.1 (Continued)

Entity	Parameter	Description	Range
Platform	Location	Position of the platform (top left)	—
	Size	Size of platform in width and height	—

Many existing Automated Game Design systems generate large volumes of candidate games and use automated evaluation to choose a subset that are judged to be the most enjoyable for human players [483]. Among these, it is common to use data from AI play traces to inform the objective function (e.g. [482, 284, 55, 89]). This approach has also been used extensively for game balancing, where the game is mostly fixed except for optimizable parameters [31, 506, 378]. Another approach involves leveraging learning techniques that seek an optimal gameplay strategy, as in [543], although these methods are likely too computationally costly to have in an AGD evaluation function.

A wide variety of desired characteristics have been considered and used in fitness functions for automated game design. In addition to simple fairness, the existence of multiple competing strategies [378], learnability [482, 284], and the uncertainty of the outcome [55] are some examples. Some systems, such as ANGELINA, evolve multiple parts of the game at the same time, evaluating each part individually, as well as the interplay between the pieces [89, 90]. Our work lies somewhere between balancing and game generation: while the base game mechanics are not subject to change, the level geometry is evolved alongside the characters, and the parameter space for the characters is large.

While automatically evolving games are common, it is less common to empirically evaluate the fitness function. The success of the generated games does not necessarily mean that the fitness function is actually optimized for the desired qualities. Some prior work solves this problem directly using player preferences in the fitness function [86, 180]. This solves the problem that the fitness function may be misaligned with player judgment, but requires users to manually play and evaluate many games. Some notable prior work does take an extra step to validate their evolutionary strategy. In [55], Browne and Maire validate their fitness function with a user study that ranked the games, showing how well each criterion in the fitness function was correlated with human enjoyment of the games. In [210], Isaksen et. al. optimize directly for a target difficulty of a one-button single-player game. They conducted a user study to validate their results, showing that human judgment broadly aligns with AI judgment in terms of the difficulty of the games. However, the AI in question was a player model that makes human-like mistakes and it might be infeasible to create an accurate model of human play for more complex games. Without such a model, it is still possible to use more rudimentary AI techniques to playtest a candidate game. However, these playtests may be very different from actual human play in crucial ways, and understanding these differences is necessary to designing effective AI playtesters.

8.3 Brawler Generation

We used the Unity game engine to implement a brawler game generator BrawlerAGD [2]. We used free assets from Kenney licensed under CC0 1.0 [3]. All games can be played with either a keyboard or Microsoft XBox controllers.

8.3.1 Base Game Mechanics

Each generated game consists of two characters, a move that each character uses, and a stage on which the characters play. During gameplay, each character has 3 stocks (or lives), and there is a static "blast zone" rectangle, which causes characters to lose a life when they leave it. The loss condition is static: a player that loses all of their stocks loses the game.

Characters are parameterized by statistics used by other brawler games. Each character can jump twice, but the height of each jump is controlled by the generator, along with other physical attributes like the character's size, movement speed, falling speed, and weight. The sprite is chosen randomly and scaled according to the generator. To create nuanced attacks, the generator controls the timings of the attacks, the placement of the attack relative to the character, and the knockback direction and damage of the attack. Finally, the stage is a list of platforms that each character can stand on and block the characters from moving through them. The space of characters is large, with each character-move pair consisting of 24 independent parameters (Table 8.1) each of which has a predefined range of valid values. Stage designs can consist of up to 10

[2]Code for this project is available at `https://github.com/smshields/BrawlerAGD`. All trial games used are playable in the repository, and the full evolutionary process is available for interested readers to generate their own new games, and an example screenshot is shown in Figure 8.1.

[3]`https://www.kenney.nl/assets/bit-pack`

separate platforms.

8.3.2 Game Fitness

We perform an evolutionary search over the space of candidate games. Stages are initially designed with an algorithm that creates a symmetric stage in which platforms are placed according to the jump height of the player. Players are initially generated by generating each parameter according to a uniform distribution on the valid set of parameter values.

We evaluate each game using the results of an automated playtest. The playtest is performed by a decision-tree AI. The AI continually tries to move towards a position that would put its opponent within the hitbox of its attack. Randomness is added to character movement (e.g. changing directions unexpectedly) to ensure that separate trials of the same game are not identical and so that characters do not get "stuck" when separated by platforms. If its character is off-stage, the AI switches to a recovery behavior tree, where it tries to get back onto the stage. This AI is capable of playing interesting matches, but the games it plays are visually distinct from human play. For this implementation, we determined that this simple agent would suffice in eliminating most extreme edge cases of generated characters: overpowered characters show imbalances in damage distribution between players, and underpowered characters lose all lives or inflict no damage to their opponent.

After the automated playtest, the objective function is calculated from a variety of information. From the playtest, we obtain ℓ the game length in seconds, d_1, d_2, the total amount of damage dealt by player 1 and player 2, s_1, s_2, the number of stocks

remaining for each player after the game has ended and h_1, h_2, the total number of hits taken by each player. From this we calculate many fitness variables, each intended to influence a specific aspect of the candidate games. These variables are normalized to ensure that we do not overtune for any given property. The overall fitness function is the sum of the fitness variables.

We want games to last an appropriate amount of time and remove games that are unplayable. To keep the game "fast" paced and to keep our genetic algorithm efficient, we set a desired time and punished games that were too short or too long. We terminated overtime games, adding a constant score penalty (-35) if the games lasted longer than a minute. The time fitness for a game is

$$f_{time} = ||\ell - - - \ell_{target}|| - 35 * g$$

Where ℓ_{target} is the target game length: 45s, and g is an indicator variable of whether the game ended in a draw due to running out of time after 1 minute of gameplay.

We encourage games where more damage is dealt and more total player interactions take place:

$$f_{damage} = (d_1 + d_2)/10$$

$$f_{hits} = h_1 + h_2$$

We want each life of each player to end at an appropriate time and ensure that the system cannot blindly optimize for f_d by creating a game in which players are stuck in a corner hitting each other. We set the desired damage per stock to 100, and if the total damage exceeds that, we add a linearly scaling penalty variable $f_{stocklength}$.

We also want each game to be fair. We apply a fitness penalty to games where the total damage dealt by each player is different or where the stocks remaining are different:

$$f_{damagefair} = ||p_1 - - - p_2|| * 10$$

$$f_{stockfair} = ||s_1 - - - s_2||$$

With the overall fitness function being:

$$f = f_{time} + f_{damage} + f_{hits} + f_{stocklength} + f_{damagefair} + f_{stockfair}$$

8.3.3 Evolutionary Search

We evolve candidate games with a population size of 100, a mutation rate of 0.4, and a dropout rate of 0.5. Characters and moves are crossed-over by treating them as a list of parameters and applying single-point crossover. We leverage crossover as our character and level parameter lists are not organized according to emergent gameplay output, meaning we effectively randomize the design space between games by selecting a random point in the list and combining. Stage designs are crossed-over by treating them as a list of platforms with single-point crossover. To mutate a player or move, we re-generate 5 randomly chosen parameters using the ranges of parameters used in initial

generation. To mutate a list of platforms, the entire stage is re-generated. Table 8.2 lists the specific fitness scores of the games of the pilot study as well as the average fitness scores of the generations from which those games were selected.

Table 8.2: A table highlighting data from the generations of game samples used in the study. Note that randomly generated came from the same batch of 100 games. Average fitness denotes the average fitness of all games in a generation, and average holdover fitness shows the average fitness of all games that were kept for the subsequent generation.

Game	**Number of Generations**	**Agent Fitness**	**Human Fitness**	**Generation Top Fitness**	**Generation Average Fitness**	**Generation Average Holdover Fitness**
A	315	109.32	88.26	128.27	50.12	83.14
C	98	129.47	146.29	138.47	63.56	96.93
E	224	122.23	64.53	109.52	48.67	84.44
B	0	-12.08	11.84	41.16	-10.44	2.12
D	0	-9.98	21.37	41.16	-10.44	2.12
F	0	-10.00	13.44	41.16	-10.44	2.12

8.4 User Study

We conducted a user study to understand how the genetic search of Brawler-AGD influences the emergent properties of games when played by human players. In particular, we focus on game fairness, i.e., the equal chance for each player to achieve a win given the same skill. We had users play both (1) randomly generated games and (2) games with high fitness chosen after genetic search. This study examines how players rate games from each of these groups according to the following traits: ease of learning, balance, enjoyability, understandability, immersion, quality of design, and ease of control. We also examine differences in gameplay data by recording the total number of hits, the amount of damage, the winners, and the game length. The players were asked to describe every game in 3 words and identify their favorite game. If we can show that evolved games have perceivably higher qualities than our randomly generated games, we can say that our system is successful in designing "balanced" brawler games.

We ran 12 total trials with 24 participants. Two instances of participant data (from trials 11 and 12) were excluded due to an incomplete survey. On average, our players were moderately familiar with brawlers, with one player indicating that they were not at all familiar with brawlers and five indicating that they were extremely familiar with them.

8.4.1 Game Selection

To compare the random and evolved games, we used 6 sample games. The games were tested by having the players play a survival match against each other, with

each player having four stocks per game. To choose random games, we generated and evaluated a set of 100 games, with a mean fitness score of -10.22. We randomly selected 3 games with fitness -10 ± 5 to use in the study. These randomly generated games were named B, D, and F.

Each run of the evolutionary search plateaued with a mean fitness score of 80-100 after 12 hours of computation. The fitness runs were taken consecutively. To select evolved games, we ran the evolutionary search to completion three times (average fitness above 80 for 20 generations). Within the final generation of each run, we grouped all games with fitness 100 or higher and randomly selected a game to use in the test. Evolved games A, C, and E were generated from this process. The shortest evolved game was generated in generation 98, while the longest was created in generation 315.

8.4.2 Measurements

Users started the study by playing a tutorial level in which they learned the basic mechanics of games generated by BrawlerAGD using human-defined characters. Users were asked to verbally confirm that they understood the controls and goals of the game before leaving the tutorial. The users then played each game in random order, answering questions after each game was completed.

During trials, we collected the same gameplay data that was used to evaluate the fitness function. This included the total number of hits and damage to each player, the duration of the game and the number of stocks remaining for each player. We used these data to get an understanding of how human gameplay corresponded to our written heuristic. We dropped f_{time} and $f_{stocklength}$ terms because humans play slower than our

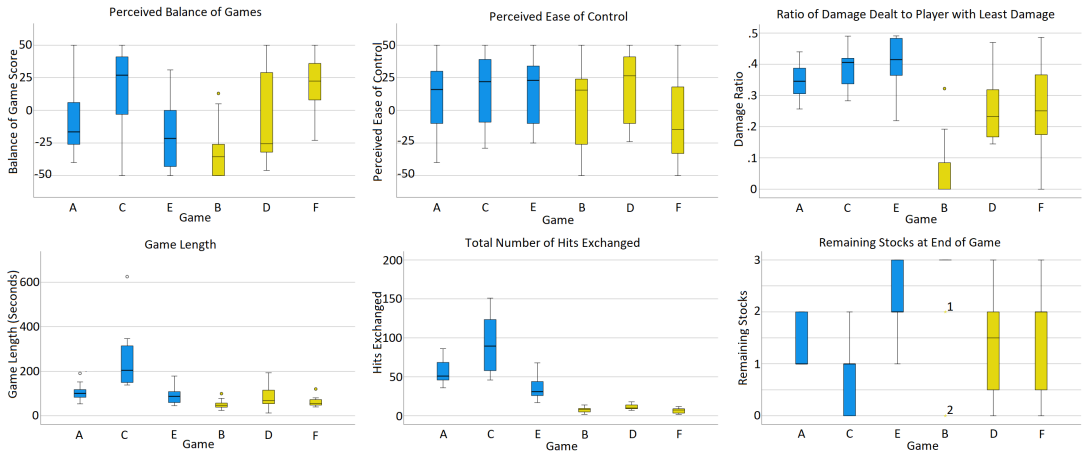

Figure 8.2: Distribution of user scores across a variety of measurement for evolved games (blue) and random games (yellow).

AI and occasionally took time to joke, ask questions, or experiment with characters.

8.5 Results

We outline the general trends across the various games, both as observed by human players and as seen through the statistical data about their matches.

8.5.1 User Perception

To understand whether evolved games had better perceived characteristics, we first had to understand if evolved game scores were statistically different from the results of the random game user survey results. We used a t-test between the two groups (evolved and random, 3 games each) to examine the perceived differences between the evolved and random games. Only one factor had a statistically significant ($p < 0.017$) difference: ease of control. Evolved games were on average slightly easier to control as perceived by the users. To understand which of the games specifically affected that

result, we performed a one-way analysis of variance (ANOVA) on all six games.

In our ANOVA comparison between all six individual games, we found that both balance and ($p < 0.001$) ease of control ($p < 0.013$) had significant differences. In particular, game C (an evolved game) was perceived to be more balanced and easier to control than the other games. Game F (a random game) was also perceived to be balanced, but was extremely difficult to control.

One simple observation comes from our post-trial survey: 63% of users listed an evolved game as their favorite game. None of the randomly generated games had a higher share of votes than the lowest evolved game.

8.5.2 Game Statistics

We also examined how human gameplay in evolved games differed from gameplay in random games, using t-tests on the quantitative data gathered from the user study. We record the damage dealt to each player, the number of hits received by each player, the length of the game, and the loser. All of these metrics, except for the game loser, showed statistically significant differences when comparing evolved games with random games ($p < 0.001$). Some examples of these trends: Evolved games tended to have higher amounts of damage and hits to each player. Each evolved game lasted, on average, more than twice as long as the randomly-generated games. These differences indicate that players had more interactions for a greater amount of time when playing evolved games compared to random games. Figure 8.2 summarizes these trends.

To evaluate fairness, we looked at the ratio of damage dealt to the player who received the least damage (see Figure 8.2). This ranges from 0 to 0.5, where 0.5 indicates

that player 1 and player 2 each dealt the same amount of damage to each other, whereas 0 indicates that one player took all damage. We use the ratio to be able to compare games in which different amounts of total damage were dealt. We found that evolved games had an on average 0.19 ($p < 0.001$) higher damage ratio, indicating that evolved games encouraged more trade of blows than random games. We found the same trend when looking at the number of hits. Despite having more activity from both players and a more equivalent distribution of attacks and damage, we did not observe that the evolved games had a more equal distribution of wins (player 1 wins as often as player 2).

8.6 Discussion

Here, we discuss qualitative observations collected throughout trials and game evolution. These trends help to provide explanations for our survey and gameplay data.

8.6.1 Game Quality

Two of the games stood out both qualitatively and statistically: Game C, an evolved game, and Game F, a randomly generated game. C consisted of one short character and one tall character, both with moves that pointed downward. In order for characters to land hits in C, they had to jump a single time, position themselves over the opponent, and trigger their attack. In F, both characters are roughly the same size but jump very high and move very quickly, making them challenging to control and land attacks on opponents. Attacks in F were high-powered and were often killed in one or two hits.

Both of these games were evaluated as "balanced" by our players by a similar margin (and these results are also borne out by the empirical gameplay data). Game C (Evolved) scored the highest on most survey questions while also producing games with the greatest length and damage/number of hits. Game F (Random) scored high in balance, but poor in control. In descriptions of the game, 3 players called the game "slippery" while others commented that the controls felt "touchy" and the characters felt "floaty". That the generator produced multiple examples of balance with human-perceived differences in game-feel indicates that the system is capable of producing balanced games that are not homogeneous or trivial in their gameplay.

C and F are just one example of a general trend in the differences between random and evolved games. Although the system can randomly create games with some desirable characteristics, it is unlikely to find games that combine multiple desirable characteristics without some kind of search. This explains the lack of statistical significance in the individual categories, despite the preference for evolved games in the aggregate.

8.6.2 Automated Playtester Alignment

Players often developed gameplay patterns similar to behaviors performed by AI during evolution. Game C scored highly during evolution as AI characters would jump over one another inside a corridor, attacking one another with their downward-facing attacks (as shown in Figure 8.1). This caused the automated testers to keep trading hits until one was flung outside of the corridor. During trials, most of the participants followed an extremely similar behavior pattern, which in turn led to similar

damage and hit outcomes.

This effect backfired when users did not follow or understand the key quirk of the evolved game. When users diverged from the most common AI behaviors, the evolved gameplay started to become very unpredictable and unbalanced. I.e., automated playtesting did not cover a broad enough set of gameplay styles to stand in for human play. Some stages forced or enticed users to play in a similar way as the automated tester, and these scored numerically higher on fitness when played by humans. The design of each game influenced how well the automated playtester aligned with human play. While the system is far from perfect in covering all gameplay possibilities, the ability for the system to even infrequently create such intricate behaviors in players by tuning parameters indicates that this style of search can create distinct, meaningful, and emergent playstyles while designing games.

8.6.3 Trends in Evolved Games

There were some general trends in game design that we observed in evolved games. Most of the evolved games tended to have characters with move hitboxes that would directly put the character in range of their opponent's hitbox. Evolved games also often had characters with attacks that could only be performed in some specific character state: for example, the downward attack in game C required players to jump first. These patterns meant that every attack would put the player at risk, which encouraged interaction between the characters. If a player missed an attack, it can be immediately punished by their opponent. Another quirk of high-fitness games is that many attacks had knockback directions that pointed diagonally backwards towards the

attacking character (for example, game E). Most human players found this knockback confusing or disorienting. Players would line up what they thought was a fatal hit, only to have the resulting knockback push themselves off of the stage. This is an example of how a fitness function designed to increase the number of player interactions can create game elements that are incomprehensible to humans.

8.6.4 Limitations

While we had modestly positive results towards evolved games in user perception, we did not show that users consistently perceived evolved games as more balanced than randomly generated ones, even if their created heuristics would have indicated to us that they should have. We attribute this in part to the complexity of the term "balance" and to making judgments about it. Although balance may be intuitive to most people, that intuitiveness does not translate to agreement on what balance is between users or games. Narrowing definitions of the emergent properties that we desire in these systems or grouping users into persona categories could help in future evaluations of emergent properties in AGDs.

The players also played against each other. When playing random games with clear mechanical imbalances (see Game B, where one player had their move behind and below their sprite), the winner would often feel very positive about their win. They might gloat, and on one occasion, one said "that was the most balanced game we've played yet". Furthermore, a player with a long history of brawler fighting games matched against a newcomer might make both feel that every game was not balanced. It should be noted that while this was qualitatively observed, the data do not clearly

bear this out—winners and losers did not have statistically significant differences in balance judgments in games.

8.7 Conclusion

We created an automatic game designer that generates brawler-fighting games. We optimized these games for simple metrics that measure desirable properties of games, such as balance, ease of control, and interactions between players. By evaluating the generated games with a user study, we learned about the ways in which our fitness function influenced human gameplay. In general, evolved games exhibited higher fitness when played by humans, indicating that automated playtesting (even with a very simple AI) can help optimize human play for given gameplay properties. The qualitative and quantitative matching of simple AI performances and human play indicate that our designer can also discover interaction patterns and mechanics that humans might enjoy.

These statistical differences did not necessarily translate well to human perception of games, a problem also described in [210]. Ultimately, the goal of evolving games towards a fitness function is to improve and shape the user experience of the person who eventually plays those generated games. Towards that end, it is important to consider how the fitness function might be misaligned with how users will experience the game. A "high-fitness" game might not actually have the properties it was optimized for when played by humans. It is also important to tune the evaluation agent to better match human behavior. A future study might look to evaluate what factors might be critical in creating a balanced game for a future heuristic, or what types of agent create the

most accurate representation of human play.

This means that data from human actors should be more central when it comes to the design of automated evaluation. One path forward is more advanced automated playtesting that incorporates user data to play the game in a human-like way. This would extend the research on player modeling in PCG such as [429]. Another method would be to change the generative techniques themselves to incorporate user data (for example, [458] does this for a single-player game). A novel approach might be to use techniques from Human-Computer Interaction, such as "Design Personas"—user profiles that guide design of an artifact [416]. Finally, by correlating design variables with human perception of the resulting game, a system could guess the perception of a novel game, allowing it to automatically adjust the fitness function in response to user feedback.

Chapter 9

FighterDDA (Dynamic Difficulty Adjustment)

> "It's an RPG. You're going to need to grind at some point; but one of the things that comes up in that community a lot is the notion of dominant strategies being too dominant."
>
> Stella Mazeika, Contributor to the Final Fantasy 4: Free Enterprise Randomizer

BrawlerAGD showed me both that procedural balancing was possible, but also that taking a naive, unrefined definition of game balance was a recipe for missing what my audience actually wanted. Were they borne of the fires of the local *Play 'n' Trade*

fight game tournaments, as I was? Or did they simply enjoy goofing around with the neighborhood crew? Such questions drove me to build the taxonomy and framework, which illuminated the diversity of player preferences that could exist when designing or tweaking any game's balance. As this image of a kaleidoscopic balance crystallized, I realized that I needed to create a new technical platform that allowed me to both test multiple balancing strategies for different audiences. Alongside this, I wanted to make sure that I investigated dynamic balancing techniques and see if a single architecture can be extended to fit multiple persona types.

FighterDDA[1] is a system built to answer a question around player-centric design—how do we proceduralize tuning for different types of players? One common trend I noticed in both game design and the academic literature was the split in definitions between competitive (equal skill means equal wins) and casual balancing (lower-skilled players have a chance at winning) strategies. As such, *Fighter DDA* seeks to investigate if we can have a standard architecture for dynamic balancing (an AI director) and leverage it to meet the needs of either of these audiences depending on the mode in which it is configured for. While an initial evaluation looked only at the outcomes of battles, I decided to try and build out ways to make the emergent, invisible elements of game balancing (the texture of gameplay) visible. Oliver Withington provided the perfect solution using expressive range analysis, and it was put to use to see the ability of the system to generate diverse sets of games for various audiences. This work presents an implementation of the framework that is more oriented around audience and evaluation—it serves to design a procedural system with a diverse set of

[1]Material on FighterDDA published at AIIDE 2025. See Shields and Melcer [432], Shields et al. [438]

personas in mind, while asking the question of how we see the results of that system before human playtesting (explored further in chapter 10.

9.1 Introduction

Dynamic Game Balancing (DGB) is the process of altering the properties of a video game during runtime to achieve the desired experience of balance of a designer for the player [16]. A subcategory of DGB is known as Dynamic Difficulty Adjustment (DDA), which focuses specifically on tuning the difficulty of one or more players during the game [26]. DDA implementations have a variety of forms, such as linear scaling of enemy difficulty (e.g., *The Legend of Zelda: Breath of the Wild* [340] where enemies spawn with more health and damage output based on the main character's progression; mechanics that aid low-ranked players (e.g., *Mario Kart 8* [114] where players in the last place receive disproportionally powerful items to help them catch up); or the AI Director pattern (e.g., *Left 4 Dead* [498] where an AI agent assigns enemy spawn rates based on a perceived player affect relating to intensity). DDA systems have been widely used in game design since the early days of the medium, and more recently have become a field of active academic study—with much research focused on how DDA systems should be implemented and how they are perceived by players [447, 388, 235, 198].

Despite their prevalence, DDA system tuning and testing is often unstructured, usually relying on designer expertise to identify what methods to use to achieve balance and how they should be implemented and tuned [217]. This is a significant issue due to the large number of parameters and methods that can be adjusted to impact difficulty

in a typical game, and due to the unpredictable combinatorial effect that changing multiple parameters could have [113]. In addition, there is evidence that the perception of DDA systems can vary significantly between audiences and players [325]. In practice, this means that balancing during the lifecycle of game development is time-consuming, occurs throughout development, and frequently requires that many playtest cycles to be properly evaluated [221, 245]. Therefore, any new research that improves or increases the ease of DDA balance could be a great boon.

In this work, we introduce our novel platform *FighterDDA* for exploring methods to overcome these challenges in game balance through the use of different AI directors (AID) and visualization tooling. Tools such as the one we present could help designers get closer to a balancing goal prior to testing, evaluate the success of a balancing approach from a quantitative perspective, and understand if a given design direction holds promise. We ground our system in the turn-based role-playing game (TBRPG) genre, using two different AIDs with varying goals. That is, the directors balance for 1) audiences of similar skill levels and 2) audiences of differential skill levels. We found that our approach of working in a simulated environment containing multiple AID implementations alongside visualization tools helped to fulfill our target design goals.

9.2 Background

9.2.1 AI Directors

AI Directors are a dynamic game balancing technique in which an agent (often invisible to the player) watches and evaluates the current game state, providing changes

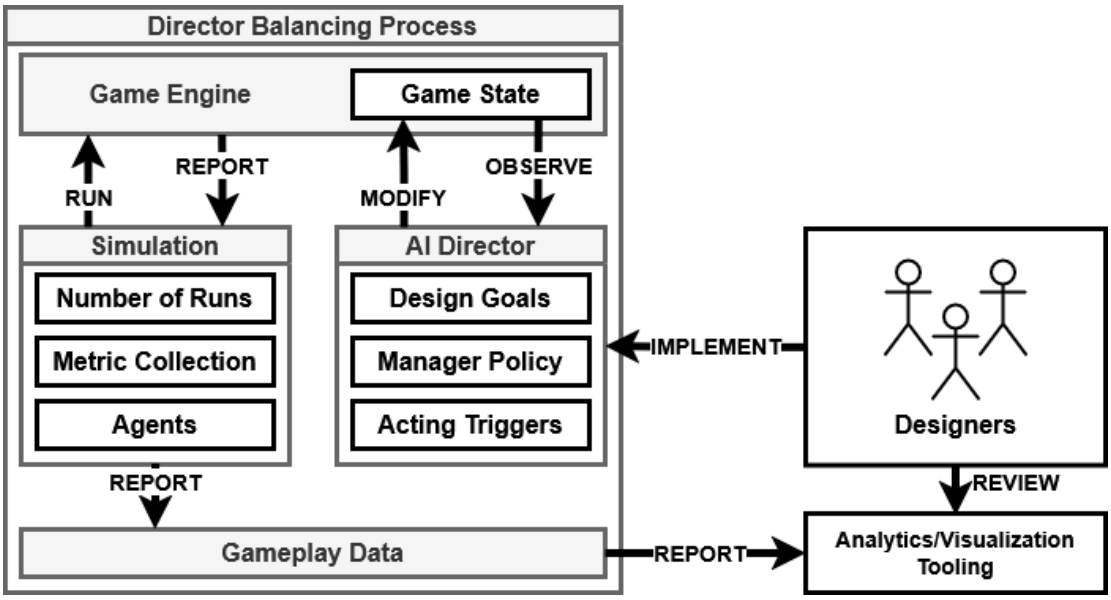

Figure 9.1: A conceptual diagram detailing the interaction between a game engine, simulation manager, director manager, data collection, and designer approaches. Our tooling suggests an iterative loop using the visualized output data from a simulation involving a director to make tuning adjustments to the director itself.

to the runtime system to change the play experience to better align with designer goals [127]. The *Hamlet* system introduced by Hunicke [198] is an early documentation of this approach, changing item allocations dynamically in response to player progression. A well-known version of this approach appears in the game *Left4Dead* [498], where enemy spawns are driven by a perceived level of intensity in the gameplay [45]. Directors can vary depending on designer goals—e.g., leveling the field for mismatched players or modulating difficulty for evenly matched ones. The key elements of directors include the game state space and tuneable parameters (what does the director evaluate and use to change game state), the manager policy (how does the director change the game state), and the opportunities to act (when does a director act) [479]. These elements can be seen in context in figure 9.1. Using a director with multiple implementation meth-

ods is present in work surrounding the testbed *FarmQuest*, which uses PaSSAGE [480] and reinforcement learning director approaches in regards to quest selection in a cozy game [251, 253]; although these works evaluate the efficacy of a single design strategy with multiple technical implementations instead of methods that target fundamentally different design goals.

9.2.2 Turn-Based Role-Playing Games

TBRPGs have been a staple of video games ranging back to the 1980's, evidenced by franchises such as the *Ultima* series [351]. Descendants of tabletop role-playing games, TBRPGs have enjoyed commercial and critical success, ranging from the best-selling *Final Fantasy* series [459] to the 2025 critical darling *Clair Obscur: Expedition 33* [417]. These games have common attributes: the player controls some number of characters with varying abilities (attack, heal, magic). Characters have some form of speed stat that determines the turn order. Characters work together to defeat enemies, who possess their own sets of abilities. Characters win by reducing their opponents' health to zero. The popularity of these games, as well as their ability to be efficiently simulated headlessly, makes them an ideal context in which to implement and test various director-based dynamic balancing systems. AIDs are underexplored in TBRPGs, presenting an opportunity to use a common yet under-examined environment to test director implementation. The adversarial play, complex action space, and simplified action queue of the genre make TBRPGs a good starting metaphor for other genres that layer additional dimensions on this framework (e.g., dexterity in action/adventure RPGs).

9.3 System Description

FighterDDA is a simulation testbed that simulates a Turn-Based Role-Playing Game (TBRPG) fight between two teams, which facilitates either a player vs. player or player vs. AI agent battle. FighterDDA focuses on the TBRPG genre, as it has a long history of success in the industry in series such as *Final Fantasy* [459] and *Pokémon* [142]. The genre is interesting for ERA analysis, as these games have a wide variety of expressivity depending on the style of combat encounter desired by the designer —easy farming encounter, mini-bosses and bosses, or player vs. player adversarial matches. FighterDDA is a headless system that allows it to run thousands of simulations on the order of minutes. It emulates a common battle system used in games such as *Final Fantasy VII* [460], where an action meter gradually fills up for each character. When the meter is full, the character is allowed to perform an action from a list —attack, magic, defend, heal, etc. Each team is initialized using a rubric of different character archetypes with unique stat and action potentials and fuzziness applied to initial stat creation. All characters have unique archetypes (warrior, mage, cleric, and rogue) that align with established patterns in the RPG space. The stats of each character determine the damage, order, and effectiveness of action. Initial stat fuzziness introduces team variation between runs. The game ends when one team is eliminated or a maximum number of actions is reached (resulting in a draw). The full source code for the simulation and corresponding tooling can be found on GitHub (a link to the repository can be found in Appendix B.3.)

9.3.1 System Goals

FighterDDA is intended to evaluate the effectiveness and game types enabled by the introduction of a variety of AI directors within the combat systems of TBRPGs. These directors have varied goals and methods to achieve these goals, aligning with earlier points in the related work section on DDA that a single game platform can contain a variety of dynamic game balancing approaches. The director follows the standard experience manager pattern —given a goal for a desired gameplay experience, an agent evaluates the current game state and then applies changes to the game world in order to meet its goals. In this paper, we evaluate the performance of three director styles in particular, with varied goals and strategies to achieve them:

- ***Inclusion*** – Aims to make games closer and therefore more dramatic for two players of differing skill levels. Achieves this through adjustment of character stats and application of targeted damage and heals. Uses one optimal and one random player agent.

- ***Difficulty-Player*** – Aims to make games more/less difficult for two players of similar skill levels. Achieves this through the adjustment of character stats. Uses two optimal player agents.

- ***Difficulty-Environment*** – Aims to make games more/less difficult for two players of similar skill levels. Achieves this through the adjustment of environment attributes such as damage scaling. Uses two optimal player agents.

The designer of FighterDDA would be able to understand the character of

fights produced by these director conditions, providing signals for what types of dramatic experience they can provide for players based on which is active. The designer should be able to understand the variation within a given designer setting, ensuring that a given setting can still produce a diversity of experiences while still achieving some specific experience goal. In the following sections, we document an approach that investigates and categorizes the expressive character of each of these director settings in order for a designer to answer such questions about the range of possible play-traces in the FighterDDA system.

9.3.2 Simulation Design

The system simulation runs the game in units called "time steps". Each time step, the character speeds are evaluated to see if they have reached a threshold where an action can be queued for execution. If multiple characters reach the threshold at the same time, the ties are broken by how much the threshold was broken and then by the base character speed. If all of these are tied, action order is picked randomly. When a character action is ready to queue, a utility agent is used to select which specific action should be performed based on the game state. Each team has an independent agent controlling it, and utility scoring is concerned with eliminating enemy characters while preserving the health totals of the ally. Once an action is selected, it is added to an execution queue, which applies the action at a specified future time step. Simultaneously and at a regular interval, a separate director agent evaluates the current state of the game and decides the magnitude and style of the balancing action to perform.

The director agent currently operates in three modes, i.e., **inclusion**, **difficulty-**

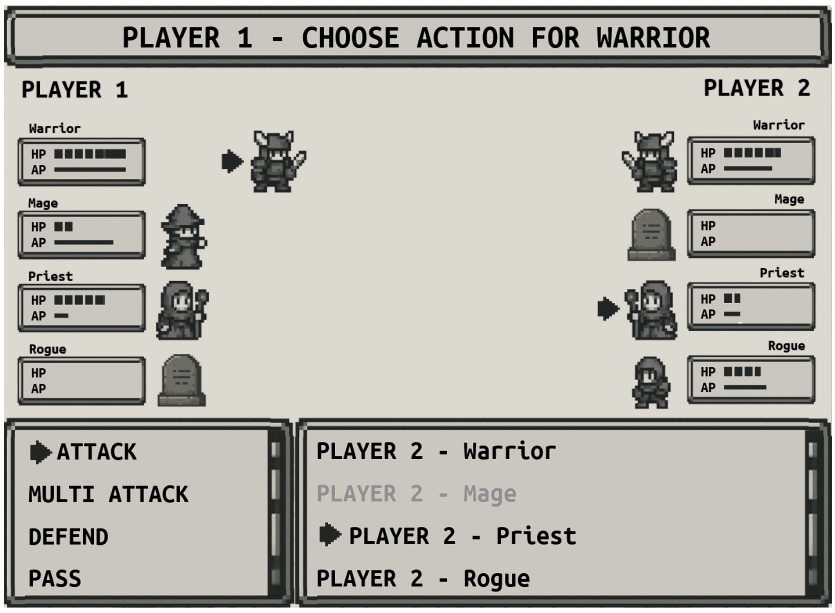

Figure 9.2: A mockup of the frontend of FighterDDA.

player, and **difficulty-environment**. These modes are described in further detail in the following section. Player agents also have different operation modes that can be used to emulate players of different skill levels. Time steps are repeated until all characters on one team have been defeated or the game reaches a maximum time (approximately 30 minutes of human playtime). A diagram representation of the simulation along with the interaction with data visualization and designer input is shown in Fig. 9.3. A visual mock-up of what the front end of the system will look like is shown in Fig. 9.2. A more detailed description of the system's design, agent behavior, and initial testing can be found in [432].

Automated playtest agents are controlled by utility agents as they are a pattern with demonstrated success in industry [477]. These agents evaluate the known game state (excluding hidden opponent data) and decide what action to take for each character

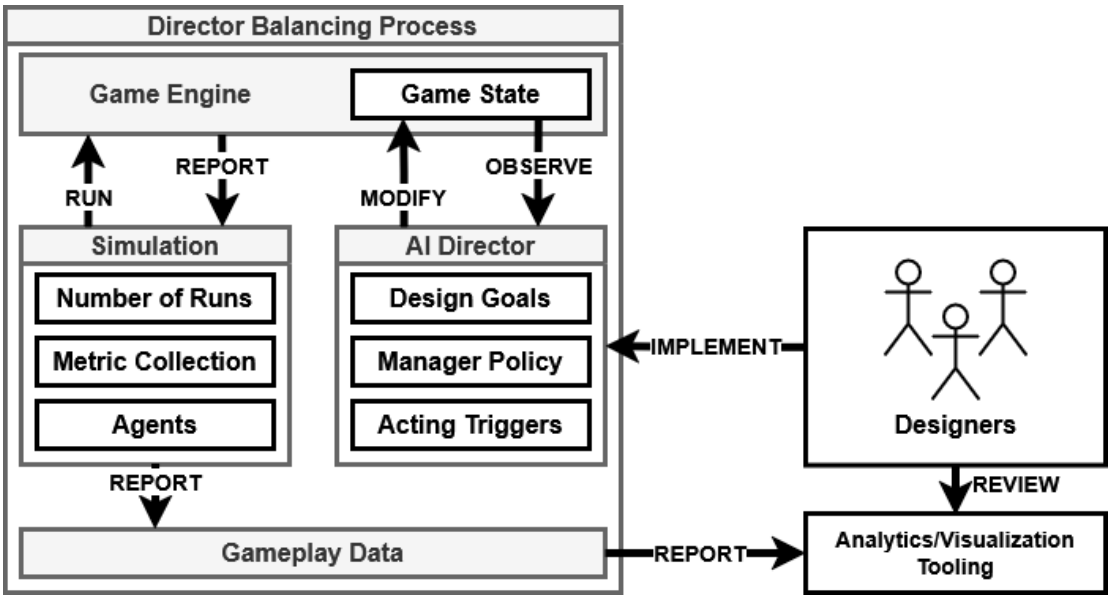

Figure 9.3: A diagram showing the core loop of the FighterDDA system.

they control. Actions are scored to prioritize defeating the opponent while preserving team health. A weighted probability based on utility scores is used for the final action selection. Within the current system, there are three agents that emulate different players:

1. **Optimal**—Selects higher-value actions. Emulates an experienced player trying to win games decisively.

2. **Random**—Selects actions randomly. Emulates a new player. Consistently loses to an optimal player.

3. **Griefer**—Selects lower-value actions. Emulates a player attempting to diminish play experience. Stalls games without attempting to win.

9.3.3 AI Director Design

The directors are state-based goal-based agents that seek to accomplish a designer goal. In our simulation, we implement two such possible goals for our directors to operate on. In one condition (differential skill level), we seek to level the playing field for players of different skill levels in a multiplayer environment. The director accomplishes this by examining the stats of both teams independently, projecting potential damage trends as a result of stat changes, and then applying stat increases to the losing team and/or stat decreases to the winning team. This aims to close the win/loss gap between players. In the second director condition, we aim to modulate the overall difficulty of the game for one or two players by adding environment-oriented changes to scalars used in damage and healing calculations. This resembles environmental effects like "fog" that reduce damage output. This director monitors team health trends, fits a line to recent data, and compares it to a designer-defined ideal trend line. The distance between the next predicted points from the line of best fit and the ideal trend line determines the magnitude and direction of the director adjustment.

9.3.4 Data Logging and Visualization

Simulations can run and evaluate games in large batches up to approximately 10,000. Each game outputs a JSON file that includes starting parameters (e.g., targeted difficulty), logs each time step with actions, director interventions, character stat changes, and summarizing metrics. Separate logs provide a human-readable summary (also viewable in console; see figure 9.4). These logs are then parsed by a graphing utility (see figure 9.10) that enables turn-by-turn analysis of a game along with important sum-

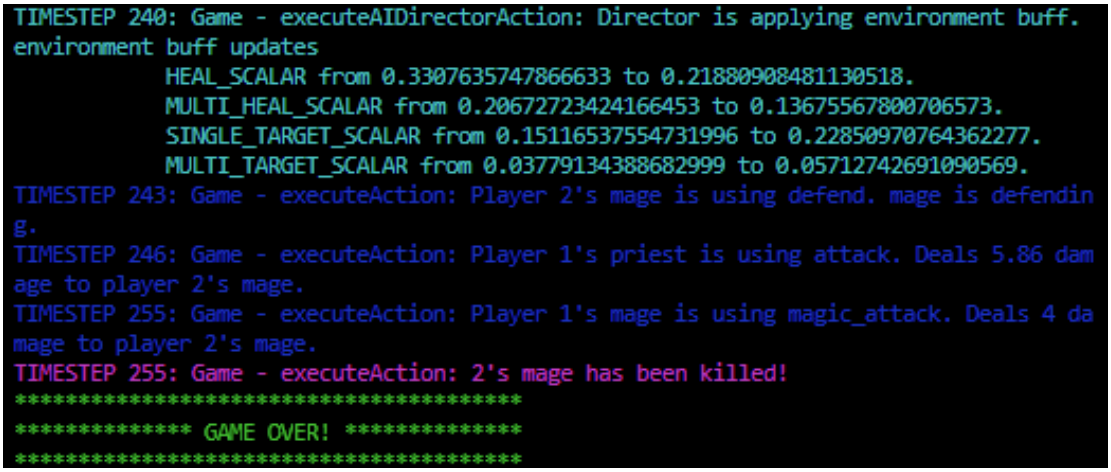

Figure 9.4: Console output detailing game actions and results.

mary metrics. The visualization, JSON game files, and overall simulation data output provide ample material for a designer to review how well a given AID implementation worked.

9.3.5 System Limitations

There are limitations that could be solved with iterative development on the platform. A playable version of the system would enable two types of user studies: player evaluation to understand if the different balancing approaches are successful for their intended audiences, and a designer survey to assess the usefulness of the tool. Implementing dynamic agents that learn and change skill level over time could indicate how transitions could be made between director modes smoothly, helping model pivotal points to move players between different models and serve different gameplay experiences accordingly. The directors have simple, heuristic-driven strategies that could also be improved by having a more nuanced or dynamic approach to tuning game elements.

9.4 Evaluation

Our evaluation had two main stages: first, a raw data analysis was performed to understand how the different directors could close gaps in the performance of differently and similarly skilled players. After this, an ERA evaluation was performed to understand the expressive nuance of the playtraces generated by agents within different director conditions.

9.4.1 Raw Data Analysis

Our evaluation consisted of testing simulation output against two defined designer goals: 1) can a director be applied to lessen the gap between differently skilled players and 2) can a director be applied to change the overall difficulty for two similarly skilled players as reflected in game length. To answer these questions, we simulated 1000 games per director configuration. In the differential skill condition, the director significantly reduced the win/loss/draw disparity, either increasing the lower-skilled player's win rate or increasing the draw rate between players. Table 9.1 details the results. The "Condition" column details the pairing of different types of agent in a match against one another. With the director disabled, one should expect agents of the same skill level to trade wins equally and to have some measure of draws. Because "Griefer" agents attempt to extend games, we should see an increased amount of draws present wherever they play. When the director is enabled, the win rates between players of different skill levels should become closer and the amount of draws should increase, indicating that the lower-skilled agent has a greater ability to potentially win games. The director should

not greatly impact agents with the same skill level and should maintain a relatively equal win rate. We find these conclusions borne out in the table—with an interesting side result of the director extending games into draws in similar skill conditions, suggesting the "equalizing" action of the director can cause extensions in game conditions. This director behavior is desirable under similar skill conditions; we found that setting a target difficulty could significantly impact game length while maintaining consistent win rates and avoiding draws. The average number of actions per game based on difficulty setting ranged from 28.55 to 76.96, suggesting that both directors align with their intended design goals. Taken together, we find that a director with a single governing architecture but different intervention policies can cater gameplay to different personas and use cases of players.

Condition	**Director Disabled**			**Director Enabled**		
	P1	**P2**	**Tie**	**P1**	**P2**	**Tie**
Optimal, Optimal	47.5%	49.6%	3%	46.4%	45.5%	8.1%
Random, Random	42.7%	38.1%	19.2%	17.4%	18.2%	64%
Griefer, Griefer	0%	0%	100%	0%	0%	100%
Optimal, Random	93.3%	5.3%	1%	59.1%	16.8%	24.1%

Continued on next page...

Table 9.0 (Continued)

Condition	Director Disabled			Director Enabled		
	P1	**P2**	**Tie**	**P1**	**P2**	**Tie**
Optimal, Griefer	55.2%	0%	44.8%	36.7%	0%	62.3%
Random, Griefer	30.8%	0%	69.2%	9.5%	0.2%	90.3%

Table 9.1: Player Win Rates Using Differential Skill-Level Director

9.4.2 Expressive Range Analysis

Methodology

Here we present our approach for analyzing and evaluating the play experiences delivered by a work in progress DDA system. The intention is to give game and systems designers the information they need to confirm whether or not a DDA system is going to give players the intended experience or set of experiences using visualization and analysis techniques beyond raw stat reporting. Therefore, the priorities for designing this workflow were first to privilege designer intention and allow for the fact that there are innumerable goals that a designer could have for a given system and second to deliver information about the system in a way that can be easily understood in terms of these goals.

Our approach involves three primary steps which are arranged in linear series. Although this series is iterative and can be re-run at any time if a design flaw or idea is noticed and corrected. We feel this process best reflects the iterative tuning and design which is typical in real world game development, and also accommodates the difficulty of tuning DDA systems which have to support diverse players and play experiences through the balancing of many parameters and mechanics. A high-level overview of this process is available in Fig. 9.5.

While the DDA system itself is built in JavaScript, the evaluation system is developed in Python. The complete platform and all tools presented in this work are available in an online repository hosted on GitHub B.3.

Play-trace Collection To facilitate analysis, we first need to gather play-traces. FighterDDA was designed to deliver players combat encounters with autonomously controlled opposing teams which possess access to the same combat capacity and abilities as the player. To gather play-traces from the system, we tasked two autonomously controlled teams with the efficient defeat of the opposition and then recorded the results. Any time a character on either team took an action, this action and the associated game state were logged to a JSON file, enabling in-depth analysis of how the fight progressed.

We evaluated the three different modes of operation for the DDA system (See Section FighterDDA), as well as a 'No-Director' mode to explore the functioning of the system when all DDA adjustments were deactivated. For each mode, we generated 1,000 play-traces to use in the system evaluation as it was felt that this struck the right balance between exploration and speed of generation.

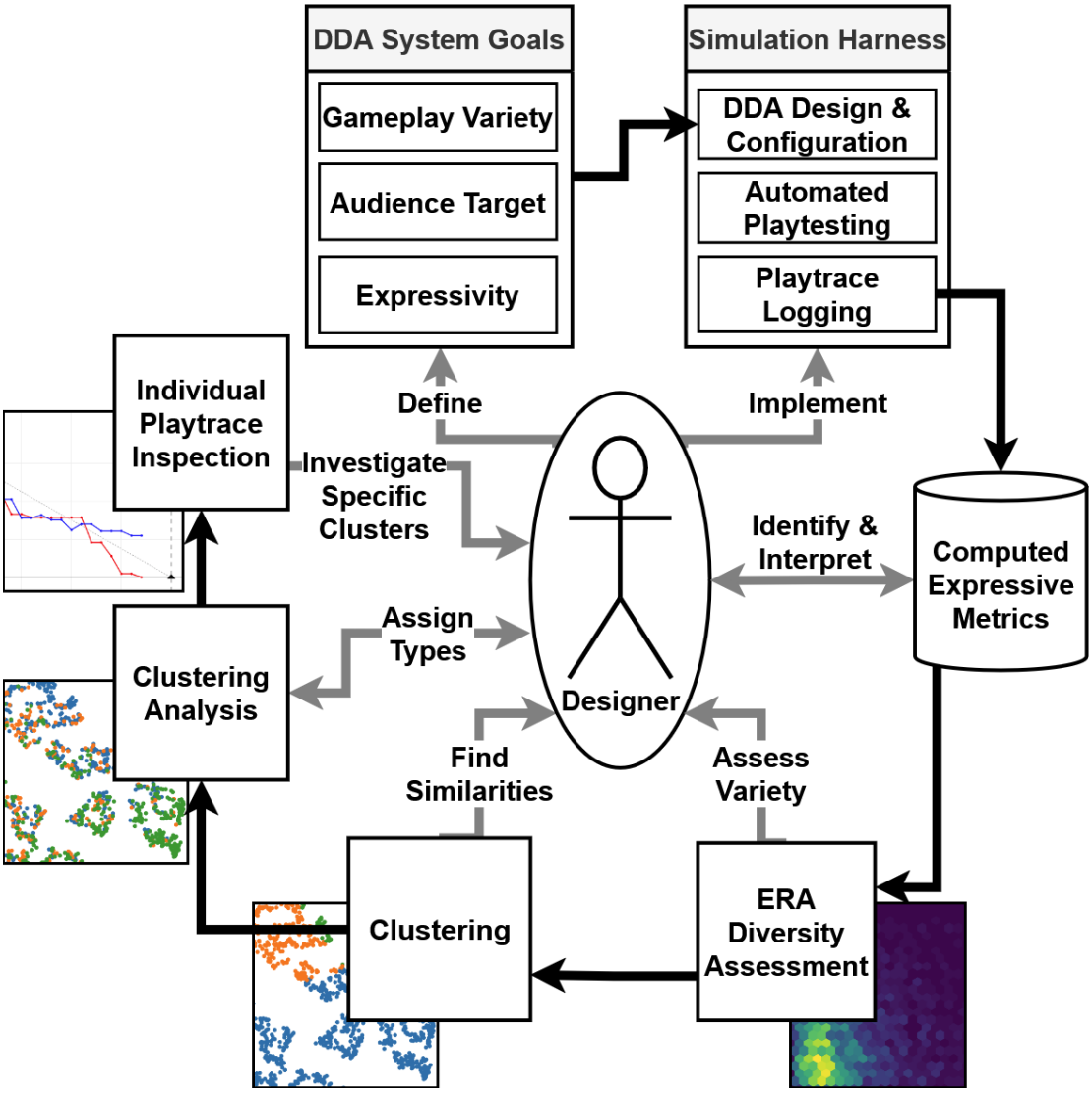

Figure 9.5: An overview of the process of analyzing the expressivity of a DDA using a simulation testbed and ERA methods. The designer is at the center of the process and can both define and derive insights from each step —they decide and evaluate metrics, get insights on gameplay variety and typing from ERA and clustering exercises, and can investigate specific play-traces.

Metric Calculation For the experiments presented here, we calculate five metrics for each play-trace, each of which is explained in Table 9.2. As ERA has not been applied to play-traces previously, there is no precedent for what metrics should be used. We instead relied on the intuition of the system designer and exploratory analysis of what appeared to give interesting clusters.

The first two metrics (Total Time Steps and Absolute Stat Difference) are focused on system functioning and are used to conduct ERA. However, the second set of metrics (Remaining HP, Damage per Step, and Lead Changes per Step) are calculated to try and capture the dramatic character of encounters and are used in unsupervised clustering.

Metric Name	**Description**
Total Time Steps	The length of the encounter.
Absolute Stat Difference	The difference in performance stats between teams at beginning of game.
Lead Changes Per Step	Average amount of times the team with the highest percentage of total health remaining per time step.
Damage Per Step	Average amount of damage done by both teams combined per time step.

Continued on next page...

Table 9.1 (Continued)

Metric Name	**Description**
Winning Player Remaining HP	Remaining health of the winning team as a percentage.

Table 9.2: A list of metrics generated by game simulations and used for ERA and clustering approaches.

Step 1: ERA Diversity Assessment The first step of the analysis is to confirm whether FighterDDA can produce the desired diversity of play experiences in terms of encounter length and initial combat ability. To do this, we use Expressive Range Analysis (ERA) to visualize all play-traces in terms of Total Time Steps and Absolute Stat Difference. These were calculated to explore the distribution and bounds of each, as we want to ensure that encounters are neither too long nor too short and that teams are initially balanced enough while not being identical. We were also interested in the relationship between the two metrics. For example, a negative linear correlation between the two could indicate that the DDA system was failing to compensate for the ability difference between teams.

Step 2: Encounter Clustering The second step used in our analysis is the unsupervised clustering of metrics related to the dramatic elements of play-traces, specifically

of the three drama-linked metrics we highlighted in the above section on metric calculation. The goal of this step is to describe the types of dramatic encounter supported by FighterDDA. Whereas in Step 1 we were interested in the distribution and bounds of the metrics here we are using them to make discrete the play-traces into distinctive sets which contain a shared dramatic experience. The inspiration for this approach came from the writing of people such as [47] on the types of enemy encounter, as well as the work on drama curves.

Although there are many unsupervised clustering algorithms, we opt to use K-Means clustering, specifically scikit-learn's implementation [2], as in our initial testing it appeared to perform better than alternatives such as DBSCAN in terms of cluster robustness, but also because it is easily interpretable, with each point belonging to a single cluster. The only parameter in K-Means is the number of clusters (k), which we determined using the 'elbow method', i.e., calculation of the inertia for each possible k value and looking for the plateau point.

We then aimed to make these clusters interpretable, which we do by producing summary statistics for each cluster in terms of the drama metrics used to produce them. We calculate the mean and standard deviation for each metric for each group, as well as the minimum and maximum values. This gives us the information they need to understand what types of play-trace are found in each cluster, but the interpretation process is manual, a limitation we discuss further in the Limitations & Future Work.

[2] `https://github.com/scikit-learn/scikit-learn/blob/da08f3d99/sklearn/cluster/_kmeans.py`

Step 3: Cluster Analysis The goal of our final analysis step is to explore whether the encounter types discovered in Step 2 also contain the diversity identified in Step 1, as well as to explore whether all clusters are possible from all DDA modes. Without this step, we could miss that while the FighterDDA system is delivering a desirable set of encounter types with the correct level of overall diversity, that some of these encounter types present players with highly predictable experiences.

To assess this, we reconduct the ERA visualizations produced in Step 1, except that we differentiate them by the cluster membership produced in Step 2. In case the visualization is unclear, we also produce the same summary statistics as in Step 2 except for Step 1's metrics. In particular, Step 3 involves effectively repeating Step 1's analysis. However, we still argue that Step 1 is valuable and appropriate as a primary step in this method, as it is comparatively quick to perform and finding a negative result would negate the need for Steps 2 or 3.

Results

In this section, we describe the results that we have obtained by applying our methodology to the FighterDDA system. We do this to both demonstrate the strengths and limitations of the approach when it is used on a real in development game system, and to give a more tangible idea of how similar approaches could be applied in alternative domains.

Step 1 Results The ERA heatmap plot in Fig. 9.6 clearly shows a strong tendency within FighterDDA encounters, with a strong cohesive hotspot in the bottom left of the

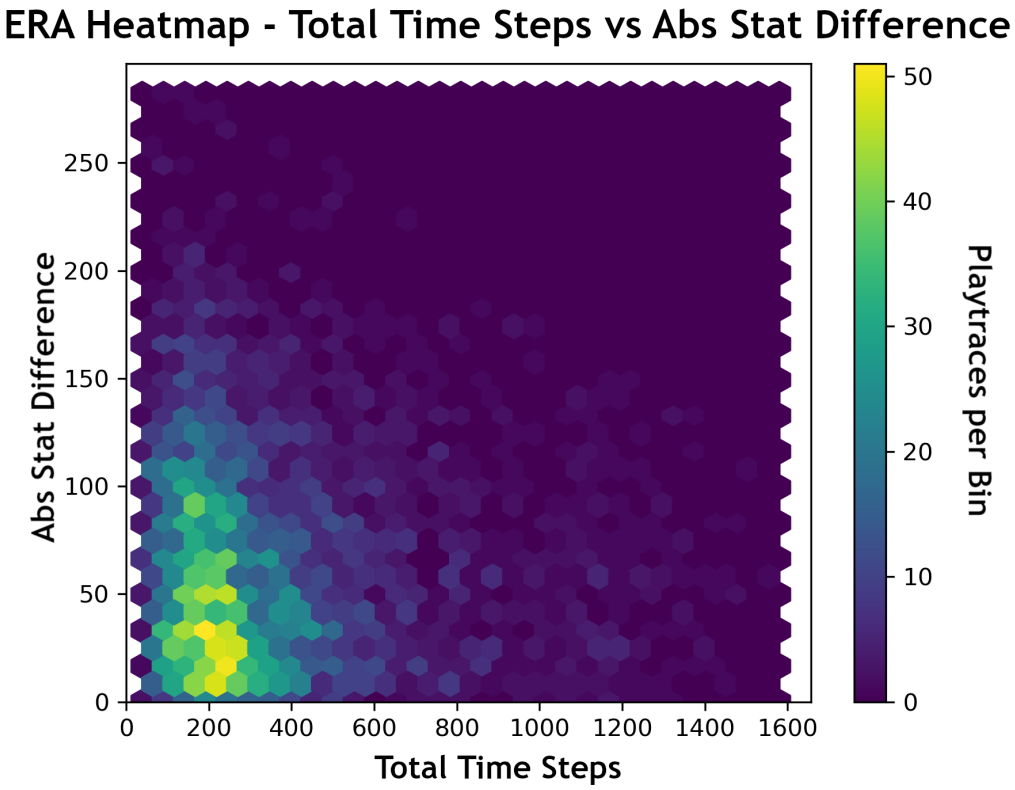

Figure 9.6: ERA Heatmap of the relationship between Total Time Steps and Absolute Stat Difference.

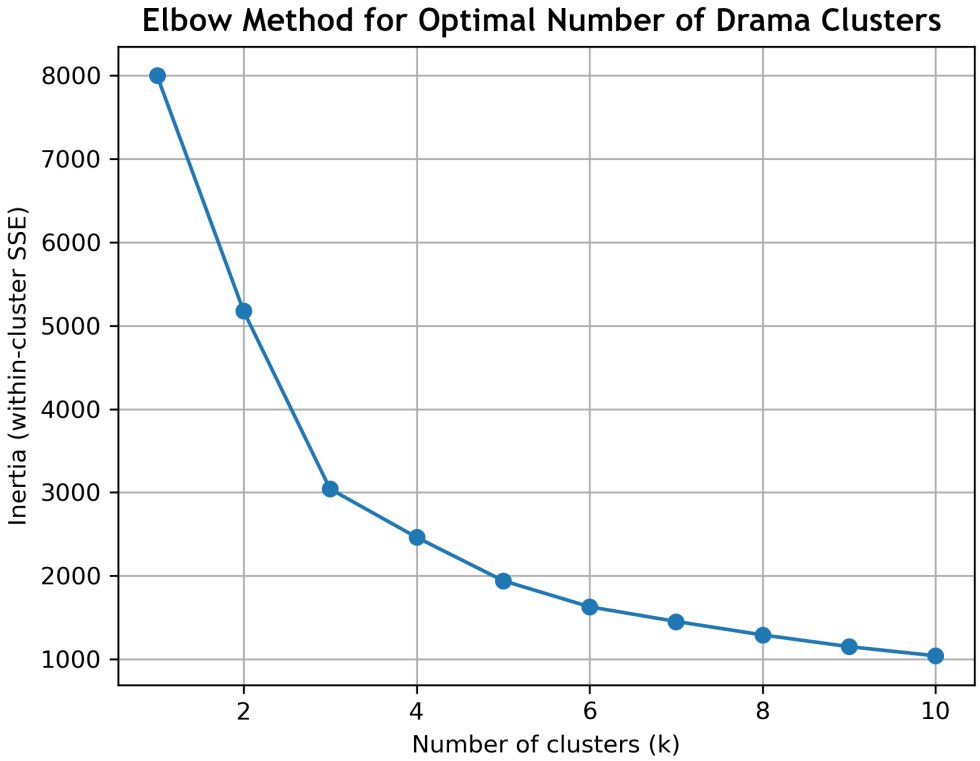

Figure 9.7: Visualization of the inertia of various K-Values when applied to the drama metric dataset.

plot roughly between 50 <Total Time Steps <250 and 0 <Absolute Stat Difference <70. It also shows the extreme variance in the outlying portions of the expressive space, with a handful of encounters being much longer or with very mismatched starting stats.

Step 2 Results

Cluster	Metric	Mean	Stddev
	Winning Player HP	**173**	36.8

0 Continued on next page...

Table 9.2 (Continued)

Cluster	**Metric**	**Mean**	**Stddev**
	Lead Changes Per Step	0.017	0.0101
	Damage Per Step	2.49	0.901
1	Winning Player HP	117	51.1
	Lead Changes Per Step	**0.0498**	0.0221
	Damage Per Step	**7.77**	2.79
2	Winning Player HP	69.5	36.3
	Lead Changes Per Step	0.0273	0.0134
	Damage Per Step	1.85	0.759

Table 9.3: The mean and standard deviation of the metric values for each of the three clusters found through applying K-Means to the drama metrics. The highest mean values are highlighted in bold.

The K-Means elbow visualization in Fig. 9.7 appears to show plateauing at

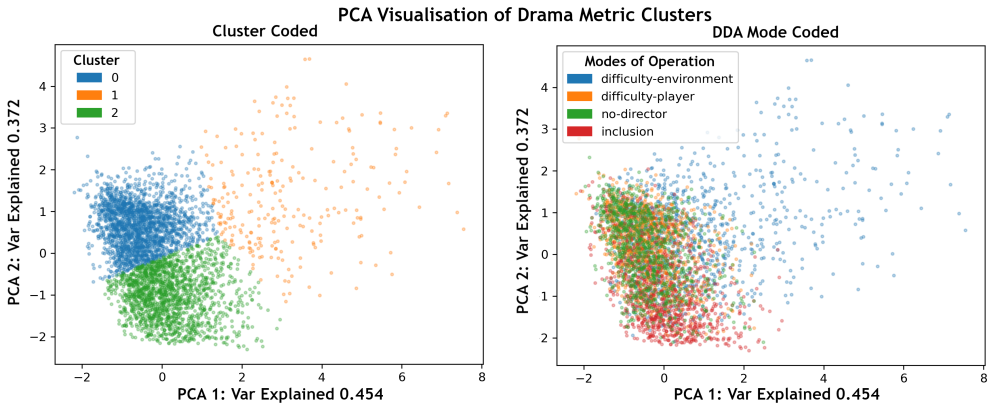

Figure 9.8: Visualization of the first two Principal Components derived from applying PCA to the set of drama metrics. Color-coded according to cluster (left) and DDA mode source (right).

K=3, so this is the number of groups we opted for. However, we should note that it is not a very strong elbow that could indicate weaker cluster membership. To further assess this, we also calculated the silhouette score of K=3, which gives a score from -1 to 1 on how strongly differentiated the clusters are. We calculated a value of 0.439, which confirms that the cluster membership is relatively weak. That being said, the clustering produced readily mapped on to gameplay styles and corresponded well with the director conditions specified (see Discussion), indicating that even with a relatively weak clustering it still produced valuable insight into the system performance.

We applied K-Means clustering to the drama metric set with k=3 to categorize all play-traces and then visualized these clusters after applying PCA to the metric set to reduce the set to two dimensions from three (Fig. 9.8, which was done while still explaining 82.6% of the variance in the underlying set. This visualization shows that there are two dense and well formed clusters in the space, labeled 0 and 2, and one diffuse cluster with lower play-trace membership labeled 1. We also contrast this in the

same figure with the same color-coded points based on the DDA mode that produced them, which shows significant overlap in all modes in their cluster 0 and 2 membership, and that cluster 1 is almost exclusively produced by the 'difficulty-environment' mode.

We also produced the summary statistics for each cluster displayed in Table 9.3. Our interpretation of the metrics gives us the following characterizations of the dramatic clusters supported:

- **0: *Grindy***: High HP remaining, low lead changes, low damage per step. Boring, low-tension, drawn-out. Comparable to a battle with a weak enemy team that has a large health pool.
- **1: *Explosive*:** Medium HP remaining, very high lead changes, high damage per step. Dramatic, tense, swingy. Comparable to a battle with two teams of equal strength and high potential to punish mistakes and leverage advantages.
- **2: *Strategic*:** Low HP remaining, medium lead changes, low damage per step. Endurance-oriented, neck-and-neck, tactical. Comparable to a traditional "boss" encounter —long games with smaller HP ratio changes over time that have a sense of tension and strategy.

Step 3 Results Finally, for Step 3 we return to Step 1's expressive range analysis but this time in scatterplot form, allowing us to color code points based on both DDA mode and K-Means cluster (Fig. 9.9). This shows that while the overall set is highly diverse in terms of these two metrics, this diversity does not translate to all clusters. Cluster 1 appears to be highly consistent in the length of encounters, being heavily clustered

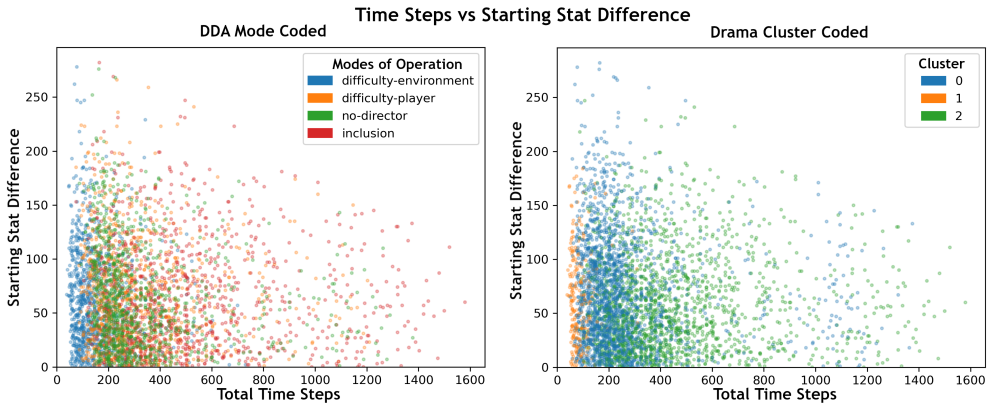

Figure 9.9: ERA Scatterplot color-coded by DDA mode of operation (left) and drama cluster (right)

between 50 and 100 time steps, as well as more clustered than is typical in terms of absolute difference. The fact that this is a peculiarity of a single DDA mode is clarified by the DDA mode coding of this set, which shows that the 'difficulty-environment' mode produces encounters that are both dramatically distinct, as highlighted in the results of Step 2, but also mechanically distinct in the time they take to conclude, as indicated by their heavy clustering to the left of the mode coded plot.

9.5 Discussion

This section discusses the system and how the ERA approach serves to help design improvements in such an automated balancing system. By doing so, I demonstrate the iterative loop of the balancing framework, showing how translation of an automated tuner produces data for an analytics process, which feeds further manual tuning from a human designer to the automated system.

9.5.1 Rapid System Improvement and Understanding

The iterative loop between FDDA and the ERA analysis proved a boon to the design of FighterDDA throughout the entire process. The production of Step 1's ERA heatmap (Fig. 9.6) helped confirm that the system had a relatively predictable game length depending on the gap in stats noted between teams while retaining the potential for "underdog" stories where one team with a higher stat difference lasts unexpectedly long or takes the win. Figures 9.8 and 9.9 show clear relationships between different director conditions and the styles of games produced, allowing easy observations of how a given director condition produced a given range of play experiences. The spread of such plots also gives a quick understanding of the consistency and variety of output for a condition—while cluster 1 (explosive) games are desirable by the designer, they are far more diffuse in terms of dramatic measurements (Table 9.3) and mostly constrained to the difficulty-environment condition. This tells the designer that this condition is doing its job to create a specific play-style, but may also create a wide variety of potential play-traces.

The plotting also allowed the designer to evaluate how well the difficulty conditions performed their job. For data collection, the director aimed to produce games of low-length between players of similar skill. In the difficulty-environment condition, this goal was achieved and achieved consistently, as shown by the vertical blue grouping of points in the DDA Mode Coded graph in Fig. 9.9. However, the difficulty-player condition seemed to fare much worse, providing a much more diffuse spread of game lengths that was potentially even less well distributed than the no-director condition.

This revealed to the designer that the difficulty-player condition was not performing its functionality as intended, even if the surface-level metric collection had initially obfuscated this fact for the designer.

9.5.2 Understanding Drama Through Automated Playtesting and ERA

Beyond testing design hypotheses and determining successful director strategies, our methodology allowed for investigation of system drama in a way that is normally only readily apparent during human playtesting. Our three clusters (grindy, explosive, strategic) are easily mapped onto various conventions in TBRPG design and represent key ways in which encounters can be shaped to impact overall player tension and progression during a game [462]. Grindy games might initially be viewed as net negative, but they are used as a pattern in games where player progression is regulated by the acquisition of experience points or enemy item drops [428]. Explosive games produce tense environments where simple mistakes can lead to major changes in game state, making every decision critical and meaningful during play. This is valuable in competitive contexts, in particular (such as the *Pokémon* [142] series' Video Game Championship competitive format [59], where the games have diverse playstyles that are both somewhat stochastic and sensitive to mistakes [17]. Strategic fights land in the archetypal "boss" category, where a longer game with subtler changes in damage and winning positions leads to a period of protracted intensity and a close finish between teams—one player *barely* survives the encounter [2].

The insight that the difficulty-environment director leads to fights of the explosive (type 1) nature is valuable and means that it would be an effective approach

in competitive environments. This fulfilled the designer's goals while providing additional positive findings—similarly skilled players (optimal agents) should have games in a relatively regular time window with lots of drama (as represented by damage dealt and changes in leading characters). Meanwhile, the lack of differences between the no-director and difficulty-player conditions indicates that this condition is ineffective in both its functional output and its generative potential.

Finally, the inclusion condition produced particularly meaningful results—not only did it have differentiation from the difficulty- and non-director conditions in how the game length was distributed, but it also appeared to have a unique dramatic profile from them, landing squarely in the strategic (type 2) bucket. This is reassuring for the goals of the condition, which seeks to make players of unequal skill levels compete on an even playing field. As noted earlier, the inclusion condition features player agents of both the optimal and random variety, meaning that without the director involved the optimal player would consistently win over the random player. As the play-traces show longer games with a small difference in remaining HP, two things become clear: First, the director is effectively closing the skill gap between the two player types. Second, the increased game length indicates that making individual mistakes would be less prone to extreme punishment, as demonstrated in the type 1 clustering, creating a more forgiving environment for players. In a player vs. AI encounter, this director would also meet goals of fulfilling the "boss" style of encounter by making the player feel as if they are always in danger of losing or winning against an opponent. [157] note that this quality of balancing such that players are constantly just-behind or just-ahead increases engagement and enjoyment of experiences overall.

9.5.3 Macro- and Micro- Visualizations in Procedural Game Systems & Generating Macro- and Micro-Gameplay Data for Balancing

The system's produced data are useful for two main reasons—it enables a macro-level view of a large number of simulations while enabling a micro-level view of individual play traces. In addition to the critical stats mentioned in the evaluation, the tooling allows designers to inspect play traces from any seed. Our initial metrics are relatively simple and do not provide a realistic judgment of if the balancing achieved actually matches the desired gameplay style (is the encounter a weak enemy, a boss, etc.). However, getting to roughly the right target provides a generated set of play traces that can be examined for specific desired patterns of behavior. For example, a designer might be seeking a sense of drama in games, looking for situations where players are constantly exchanging the lead between one another. These can be identified both by metrics and by the visual tooling provided in figure 9.10. Ensuring macro-level consistency while meeting micro-level goals provides the designer a starting point to confirm their hypotheses with human playtesters.

Understanding a procedural game system is a difficult task—while the methods to generate a game element or system might be clear, the full range of possibilities of those methods might be incomprehensibly large. Applying visualization and data-processing techniques to procedural systems helps negate the chaotic nature of testing these systems while removing the risk that a designer is exposed to misleading outliers. Being able to, at a glance, understand both the variety and dramatic character of a procedural balancing system through a visualization of a high volume of play-traces

means that designers spend less time attempting to interpret random sets of generation and instead focus on targeted changes that will meaningfully impact overall experience of their system.

A final benefit of our approach is that, as a requirement to generate the macro-visualization of all play-traces, each individual play-trace is comprehensively documented, annotated, and ready to be plotted itself. This plot is shown in Fig. 9.10, which allows investigation of any given point in the ERA plots by looking at how the leads changed, how frequently and to what extent a director acts, and so on. In this way, there is a clear pipeline to our process—after ERA and cluster analysis is complete, designers can inspect particularly interesting outliers or pick samples from clusters to get further clarification on what games of a certain type look like. This combination of macro- and micro- views of automated playtesting provides a wide range of co-creative tools to tune and optimize the exceedingly complex area of dynamic game balancing systems.

9.6 Limitations and Future Work

A limitation of our approach is our reliance on automated playtesting to gather play-traces and the unknown relationship they have with those one would get from a human survey. Automated playtesting is highly defensible from a resourcing perspective, as once the development has been done to support it it is easy to scale and costs no time or money to retest unlike human playtesting. However, it would be valuable to know how much of an abstraction this is, and in future work we intend to conduct studies

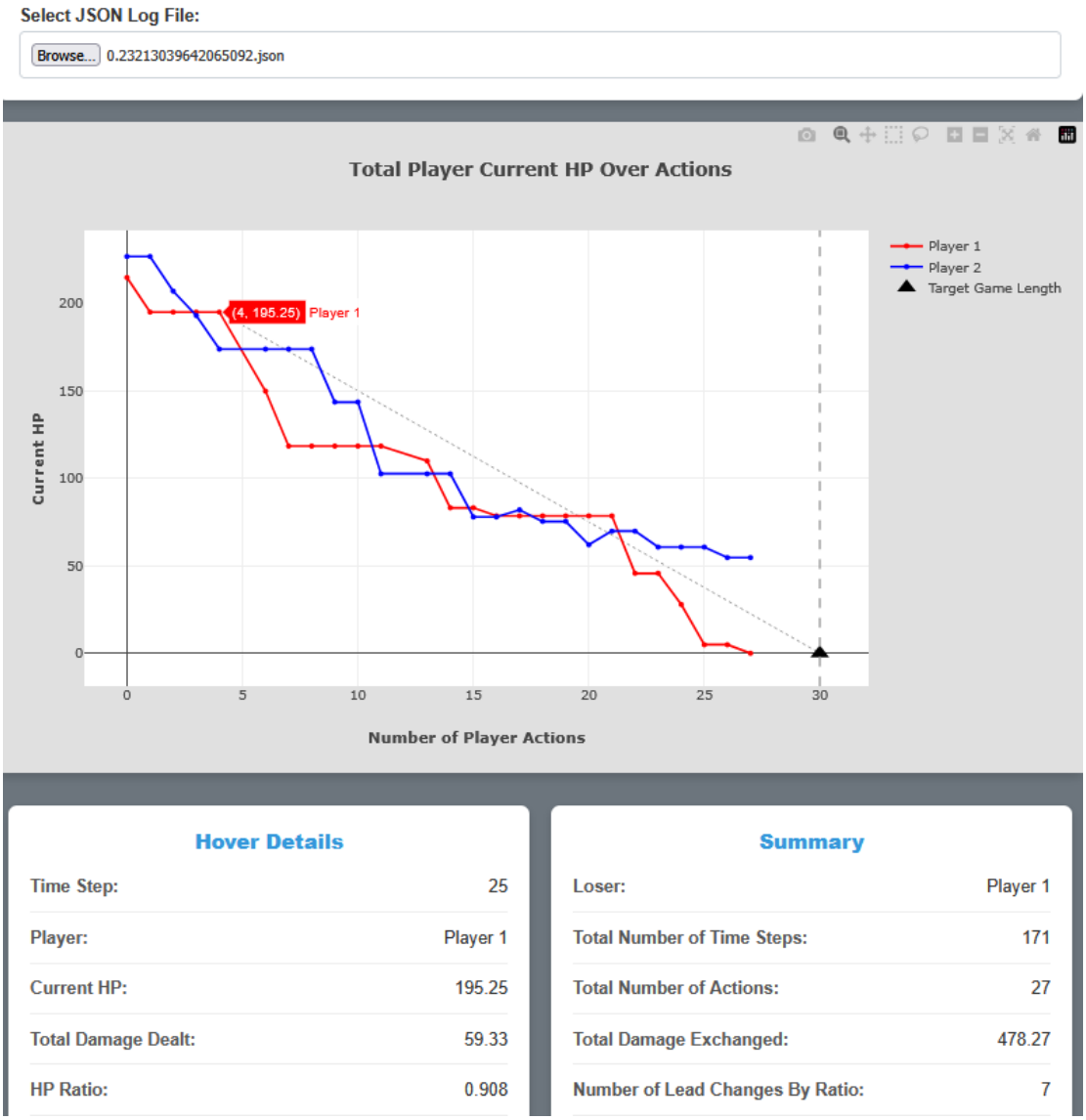

Figure 9.10: Tooling included with the simulation system to inspect individual game play traces from a batch of simulations. The graph details both players' current health, while allowing inspection of game state at each action performed. A trend line shows the slope desired for a game's difficulty (if specified).

using human players to investigate the extent to which they explore similar encounter possibility spaces to automated agents.

An opportunity to confirm our tooling approach's contribution to the MI-PCG space would be to conduct user studies where game designers are actively attempting to implement some form of DDA system and have play-traces available. In-depth case studies of our methods in more varied environments would help more concretely show that ERA can prove effective in systems-oriented contexts.

A practical limitation of our approach is the reliance on human interpretation of the play-trace clusters produced in Step 2 of our method. As we were only clustering on three metrics and only three clusters were produced, there were not too many data points to interpret. However, there is no upper limit on the number of metrics that one could hypothetically use in this process, and any increase would complicate interpretation.

Related to the above limitation, is the requirement for human designers to design the metrics of interest. This places a configuration burden on designers and also limits the evaluation to what can be conceived of and actually calculated. This limitation is harder to overcome, though we have options to explore, including ERA selection criteria [524] to decide on metric pairings for conducting ERA, as well as sourcing metrics from validated sets [299].

A study being executed at the time of this document's writing aims to present FighterDDA with a web-browser-based user interface to human players, allowing them to participate in different conditions (with director, with director targeting similar skill levels, and with director targeting differential skill levels). This study also aims to iden-

tify player personas: Do players (correctly) self identify with a given director condition, and does their enjoyment correspond with their identification? Knowing if the directors correctly address specific player personas and if players can accurately identify their own player types will prove invaluable in the future when attempting to study how to accurately tie player models to desired gameplay. The study also aims to investigate the concept of "believability" (player awareness of DDA interventions). If prior work establishes that intervention can negatively impact player self-esteem, then understanding how that intervention is detected by players can also help inform future game designs where dynamic tuning is present. To accomplish this, we will present a third-party "deity" character that can be toggled on conditions that announce balancing changes it introduces. Introducing this character in director and no-director conditions will help us understand the extent to which systematic and information-telegraphing plays a role in player's self-esteem impacts when interacting with a dynamic tuning system.

9.7 Conclusion

The importance of this work is twofold—on the one hand, AIDs can achieve diverse balancing goals through targeted implementation. However, thorough data reporting, visualization, and a large number of simulations can be used to increase the confidence that a given approach works as intended. These insights suggest how such systems could streamline the costly and time-intensive balancing process. Our cursory evaluation showed that a director framework combined with detailed analytics can support varied design goals. Investigating how directors can be designed for fundamentally

different use cases and audience of players helps clarify how analytics and visualization can inform AID strategies.

Combining reporting with player-focused AIDs offers an iterative way for designers to evaluate how their AID systems might achieve their balancing goals. We found through an initial evaluation that both diverse director approaches and thorough data reporting could expedite balancing, enabling more refined prototypes before user testing.

The outcomes of the ERA work and the capabilities of the simulation had me return to a conclusion that came from my literature review and work on a taxonomy and framework: if game balance is an experience of fairness and playtraces represent the texture of our interactions with a game, could we get an understanding of the narrative properties of game balance over the body of playtraces produced through gameplay? Such an understanding, combined with a taxonomy and procedural balancing techniques, would allow a designer to quickly build an understanding of the aesthetics of fairness that their systems possess, in turn enabling informed and nuanced tuning to meet their ideal experience of game balance.

Chapter 10

Playtrace Arc Search (PAS)

FighterDDA and the ERA analysis performed piqued my interest in the more global implications of balance design. *BrawlerAGD* showed me that balance was complex and multidimensional; *FighterDDA* showed me that balance shifts during gameplay depending on the player and tells a unique story that has fairness as the central thread. With this in mind, I wondered if it was possible to visualize such shifts and stories, and I quickly realized that progressions of balance mirrored narrative progressions and arcs brought up in many other disciplines of game design (and literature). If I could make a system that allows a designer to quickly see the development of 'fairness' throughout their gameplay, they could start to think critically about how to tell stories with the oscillations of curves produced. In other words, I wanted to create a visualization of the calculus of balance, one in which inflection points, valleys, peaks, and curves informed how the player interacts with game balance over an arbitrary length of interaction with a game system.

The Playtrace Arc Search (PAS)[1] tooling came as a result of realizing that interpreting thousands of playtraces for insights on a tuning system was going to be difficult, and while heat maps and similar analysis exist, generic approaches that could be moved between different signpost metrics rapidly were not easily portable to our environment. Additionally, deciding how to search through a large corpus of playtraces for interesting or desired examples is difficult—providing an interactive, designer-friendly method to search through such data sets (in our case, a canvas to draw playtrace curves on) provides a casual-creator friendly approach to quickly filtering through game data.

10.1 Introduction

A playtrace from a video game is defined by Osborn et al. [352] as "a sequence of player actions corresponding to one play of the game." As players perform actions in a game, observable metrics are produced as a direct or indirect impact of changing the system. Such metrics might include parameters relating to a game goal (e.g., how many items have been collected), bug prevention (e.g., has a player reached a checkpoint in a race too early), or updating a player model (e.g., what level of difficulty the player is experiencing right now). The role of such metrics in design strategies such as Dynamic Game Balancing [16] as well as scoring games for quality in A/B testing [202] helps to underscore the utility of metric production in both design time and runtime. These metrics are frequently trivially available due to their critical role in modifying game systems, and are also useful to provide system observability of a game system over a play session. Designers have the responsibility not only to tune input parameters to a

[1]Material published in PAS published at EXAG and ICIDS 2025. See [436, 437].

game system to ensure a desired play experience, but also to utilize these observable metrics to iterate on their design strategy and confirm their hypotheses about runtime operations of games. The value of producing understandable and meaningful playtraces is, as such, a meaningful step in iterative game design—especially when it comes to human or automated playtesting.

However, playtrace collection and analysis rarely involve examining a single predetermined session of gameplay data. The tools surrounding the playtraces usually deal with aggregates of thousands to millions of traces compiled together and visualized in some format. For example, heatmaps [528, 109], player modeling [536, 276], and balancing patches [222] are just a few examples of how playtrace composites are currently used to augment design strategies. While such tooling is valuable for pattern identification in playtrace datasets, it can put the designer's understanding of their desired, emergent play experience in the back seat to a global summary of data. Furthermore, such approaches might make it challenging to identify interesting examples of playtraces that could become meaningful case studies for future design investigations. As such, a tool that allows a designer to show what their intended experience is for a set of playtraces and to understand how consistent (or, if desired, diverse) a set of traces is quickly could provide a valuable strategy for rapidly understanding system characteristics during gameplay. Supplementing this with the identification of important examples of their desired experience then provides such an approach with the opportunity to do deeper investigations of important games played and even perform parameter tuning based on a given playtrace match.

To address this gap in designer tooling, we introduce the Playtrace Arc Search

(PAS) tool, which enables the searching of a large corpus of playtraces according to a user-defined curve or "arc". PAS reads a set of structured playtrace data, has the user select game and/or system metrics to analyze, and then allows the user to "draw" a curve to search for playtrace arcs that fit their arc. It also allows a designer to quickly understand if a set of playtraces exhibits similar or diverse behaviors. In both of these modes, PAS provides an option for rapid iterative design cycles, where the identification of some systematic arc (such as drama, emotional intensity, difficulty) allows a designer to focus on important qualities of a specific gameplay session. By quickly showing a designer where to focus, PAS can tighten the playtesting loop, which is cited as one of the most expensive and time-consuming parts of the game development lifecycle [548]. In this work, we describe the system architecture of PAS, perform a brief set of evaluations on a corpus of 1,000 playtraces, and discuss the design implications of applying this tool in both manual and procedural design contexts. A screenshot of the tool can be seen in Fig. 10.3, and the tool is available on Github (see appendix B.3).

10.2 Related Work

10.2.1 Co-Creative Tooling

Co-creative systems and research tooling involves altering a designer's workflow to include computational systems, which helps produce work with greater speed, quality, or expressive range. [278, 297, 258, 250]. Such systems often either provide active assistance in creating an output (such as in Liapis et al. [277]'s *Sentient Sketchbook*, where a user-drawn map is used to generate a level) or in scoring and evaluating

some user-defined artifact (as in Migkotzidis and Liapis [316]'s *SuSketch*, which provides predictive evaluation of a level design). Our system falls into the latter category and lands somewhere between a creative tool and a data analytics platform. By allowing a designer to specify what types of emergent outcome they desire, we afford a designer the ability to determine if their current game system is capable of the experiences they desire. If it is (or is not), a designer can then use specific playtraces to understand what parameters, tuning, and gameplay actions were critical to producing the playtrace they observed.

10.2.2 Parameter Tuning in Games and PCG

Video games are inherently parameterized systems. How high a character jumps or when an AI should perform an action (among many, many other scenarios) is all determined by a designer's decision on how the corresponding mechanic and parameter is tuned. As such, both static and procedurally generated content (PCG) depend on how a system is parameterized and tuned for a given interaction. The pursuit of automated parameter tuning and parameter tuning analysis is therefore well researched and prominent in the academic literature. Summerville et al. [465] identifies parameter tuning as one of the greatest unsolved challenges in PCG, and good approaches to parameter tuning provide the opportunity to greatly improve artifact quality. There are many approaches to performing automated parameter tuning—an algorithmic example is the usage of evolutionary search algorithms, which score and iterate a swath of parameters until some desired fitness is reached [433, 323, 268, 407]. An analytical approach is shown in Withington and Tokarchuk [524]'s approach to identify *which* parameters

and metrics might be most valuable when performing Expressive Range Analysis [455]. The importance of understanding *which* parameters and *how* a designer should tune them is thus critical facets of an iterative game design loop, especially when it comes to systematic tuning for emergent or experiential game properties such as game balance. This identification and modification of parameters in turn depend on the quality and observability of system metrics that tell the designer how a game progressed—metrics which are commonly reported through playtrace data.

10.2.3 Playtrace Analysis

Playtrace collection and analysis have been a critical area of game design and research for well over a decade. Drachen and Canossa [110] make the case that gameplay metrics provide critical information about player behavior and that any action a player can take can potentially be measured. Wallner [509] discuss the importance of large playtrace and player metric corpora, especially when it comes to the production of visualization tooling such as heatmaps. However, such large sets of playtraces are inherently difficult to parse and nearly impossible to investigate on a case-by-case basis [286]. This does not mean that investigating an individual playtrace is unimportant—the study of individual games between players has historically been critical to the study of games such as *Chess* [133] and *Go* [393] and is a common attribute of eSports (such as in *Starcraft* [40]) and professional gaming commentary [37, 281]. The ability to understand trends at large in a set of playtrace data and identify critical examples to investigate provides a solid foundation for thorough analysis of playtesting data.

10.2.4 Player Modeling

Player modeling refers to the practice of making a system-defined model of how a player might behave, feel, or think before, during, and after gameplay [536]. Player modeling is key to understanding how a system should respond to the input provided by a given audience, and is a common practice during game design [111] and the construction of procedural systems [362]. Modeling is particularly relevant in adaptive game systems such as dynamic difficulty adjustment (DDA), where modeling perceived difficulty allows a designer to appropriately modify a game's properties to meet a player on their skill level [26]. This is exemplified by Valve's *Left 4 Dead* [498] AI director (AID), which uses a calculated metric for "emotional intensity" as a means to define gameplay curves. In pursuit of roughly sinusoidal cycles of rising tension, climax, and falling tension, system perception of emotional intensity drives enemy spawn volume and timing [45]. It is important to note that player modeling is often a designed composite of many gameplay metrics. In the case of *Left 4 Dead* [498], there is no raw value measurement of "emotional intensity" transmitted from the player's limbic system; it is calculated from a combination of in-game metrics. Thus, we can easily visualize and operate different dimensions of player modeling through these composite metrics that are created by a game's designer. The ability to understand if your player modeling is 1) correctly tuned to player experience and 2) allows for the correct manipulation of system interaction is, in turn, critical for a designer to meet their intended experiential goals. A generic version of how a modeled metric changes throughout a playtrace can be seen in Fig. 10.1.

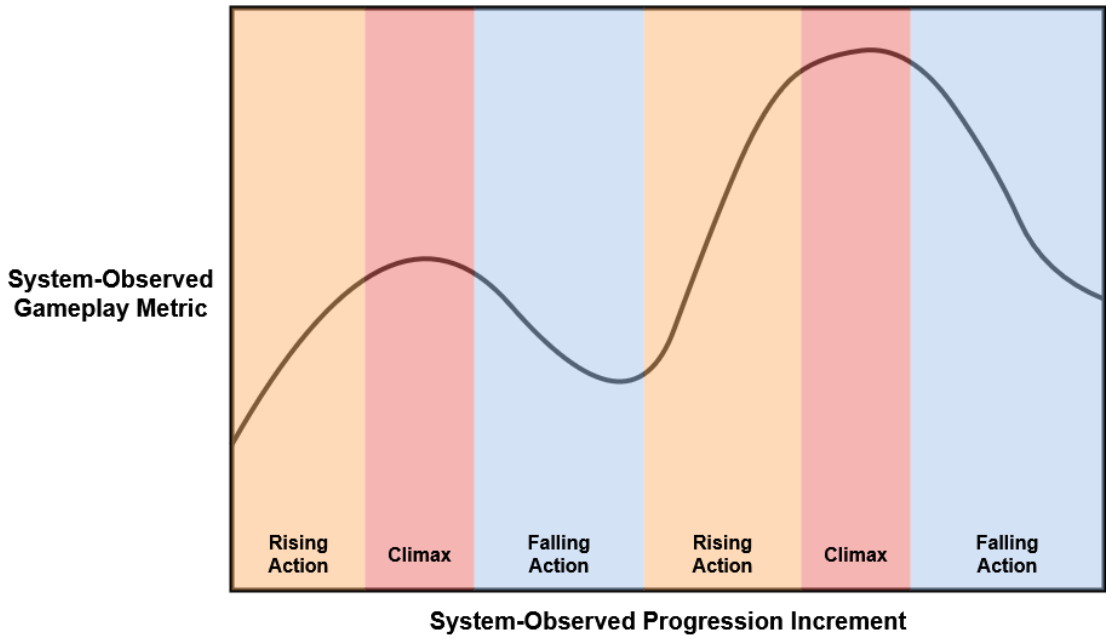

Figure 10.1: An example of how playtrace metrics are modeled through curves and used to influence designer understanding and gameplay systems. Some system-observed metric, usually derived from a player model, is meant to change over some incremental metric (e.g., time or number of games played) in order to understand, project, or modify a player's experience according to some curve. Such models have been used to great effect in Dynamic Difficulty Adjustment systems, such as the AI director in *Left 4 Dead* [45, 498]

Dramatic Arcs

Dramatic arcs represent the two-dimensional, rising and falling nature of a narrative plot over time. Frequently framed in terms of the tension of the story or the emotional character of the story, such arcs date back to Aristotle and are a critical tool for understanding the plot structure of the narrative through a visual lens [54, 22]. The representations of arcs take many forms: directed graphs [361], simple pyramids [534], two-dimensional waveforms [508], among others. These arcs were observed to be able to be computationally processed and manipulated by Kurt Vonnegut in 2004 [508], and this exercise has been carried out in works such as (Reagan et al., 2016)'s exercise in analyzing a large corpus of narratives by sentiment [383]. Their processing yielded the insight that emotional arcs generally possess some configuration of six unique waveform patterns—"rags to riches", "riches to rags", "man in a hole", "Icarus", "Cinderella", and "Oedipus". Fig. 10.2 provides a visual representation of these patterns. While not all authors define arcs according to these six labels, such labels prove useful in providing a comparison point of arcs to a common corpus of narratives [514]. Dramatic arcs have been a frequent target for drama management systems, especially since they are employed in both interactive digital narrative and in games [396, 431, 334]. Dramatic arcs are not limited to written literature, but can exist in all forms of media; how they are conveyed depends on the affordances of the media in question.

Game Systems as Narrative

Video games afford a wide range of tools to convey dramatic arcs—audiovisual feedback, gamefeel, controls, progression, script, cinematography, etc.—the list is long

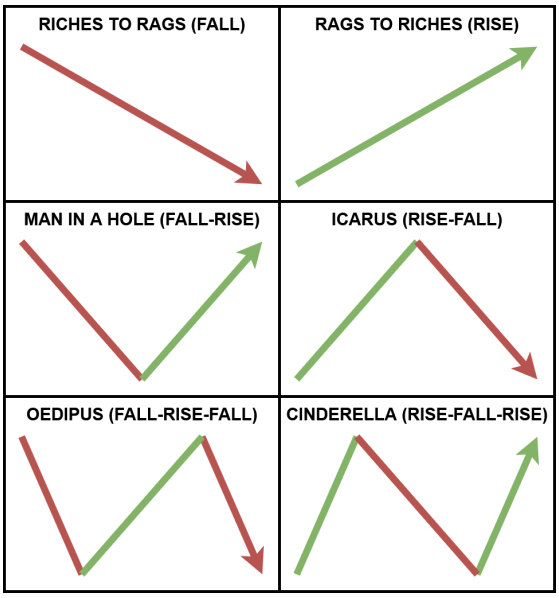

Figure 10.2: A visual representation of the six types of dramatic arcs found in narratives as defined by Reagan et al. [383]. All x-axes represent story progression, while the y-axes represent emotional sentiment.

and varied [230, 42]. The translation of these interacting systems and components is commonly understood through design frameworks such as "Mechanics, Dynamics, Aesthetics", where system definition, the emergent properties of the system, and player interaction direct the aesthetic, emotional, experiential qualities of a game from a player's lens [199, 266]. Tracy Fullerton [138] directly ties dramatic arcs into the interaction with game systems, noting that the uniqueness of these arcs in games comes from player skill and agency: "...success or failure is in the hands of the player. It is the player who must learn how to avoid the attacks, moving closer and closer to the goal." Gillian Smith et al. [456] use the metaphors of "rhythm" and "beats" when discussing content generation in 2D-platformers, showing how gameplay is often interpreted through the lens of the expressive terms of other mediums. With such a wide variety of interactions available at the hands of designers, understanding how each uniquely designed system generates

dramatic arcs is a daunting task, which requires an understanding of their intended audiences and the way their system behaves over some progression.

10.3 System Description

The Playtrace Arc Search (PAS) tool (Fig. 10.3) aims to help designers accomplish three goals:

1. Understand the overall distribution and consistency of playtrace data
2. Search large sets of playtrace data for desired system arcs
3. Inspect individual playtraces of interest, with references to their source data

PAS aims to satisfy these goals for a specific drawn playtrace within seconds, allowing multiple searches to be performed in quick succession. It accomplishes this by parsing a large set of structured JSON data and providing a point cloud representation of user-defined properties of said JSON data. The user can enable or disable this cloud on top of a canvas, which allows the user to draw a desired curve that they would like to search for within the playtrace corpus. After selecting search strategies using curve-matching approaches, a list of similarity-scored results (Fig. 10.4) is presented for the user to inspect. The system is web-native and uses graphing (Chart.js[2]), canvas (Fabric.js[3]), Fréchet Distance (Curve-Matcher[4]), and Dynamic Time Warping (Dynamic-Time-Warping[5]) libraries for user-input, visualization, and search strategy. The full code and demo for PAS is available online (see appendix B.3.

[2] `https://github.com/chartjs/Chart.js`
[3] `https://github.com/fabricjs/fabric.js/`
[4] `https://github.com/chanind/curve-matcher`
[5] `https://github.com/GordonLesti/dynamic-time-warping`

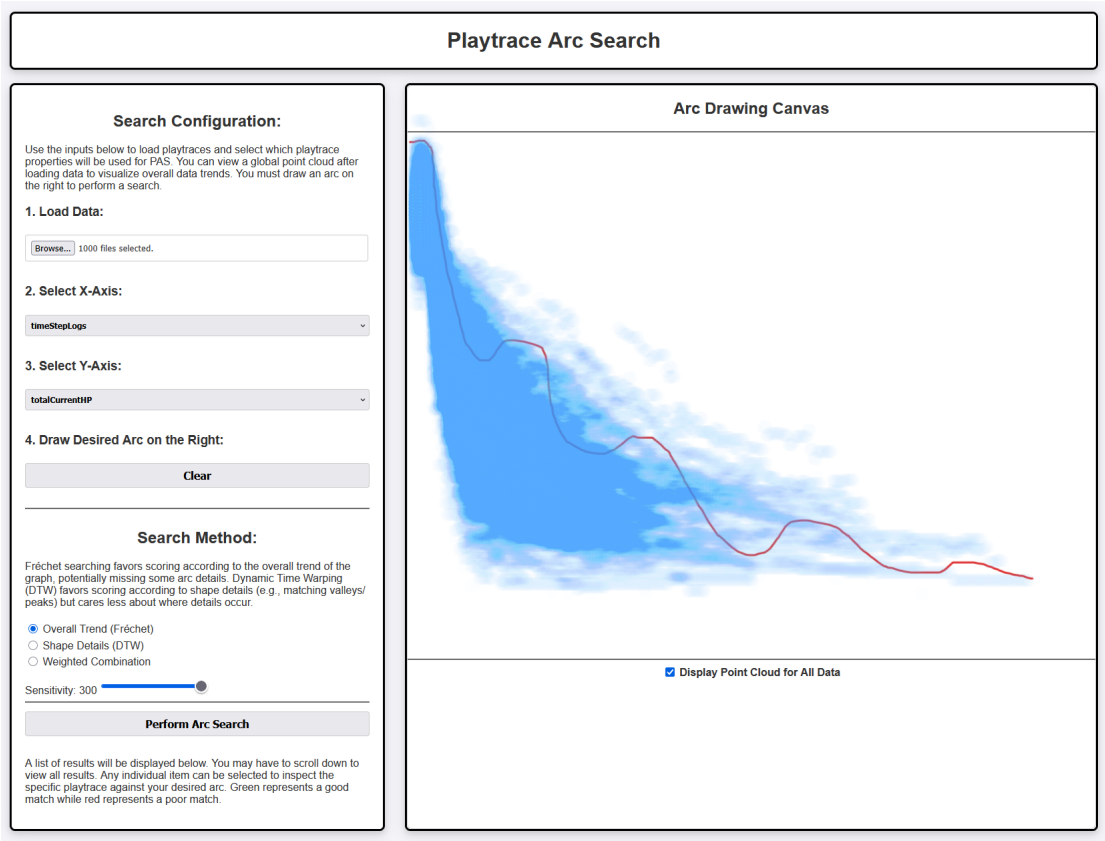

Figure 10.3: A screenshot of the Playtrace Arc Search (PAS) tool. The left panel allows a user to upload data, select the values to inspect, and the curve-matching strategy to use when evaluating playtraces. The right panel provides an (optional) point cloud graph of all playtraces, as well as a canvas for a designer to draw the curve (in red) that they are searching for in their data.

10.3.1 Metric Selection, Logging, Formatting

PAS focuses on the analysis of how important gameplay metrics change throughout the playtraces. These metrics are sometimes raw data (e.g., how much HP does a character have), but can also be productively calculated and composited into player-modeling metrics tied to player actions or progression. The metrics recorded at each step can be thought of as dependent variables, which PAS visualizes on the y-axis of its arc drawing. On the x-axis, a discrete progression measurement is used to map each consecutive data entry (e.g., gameplay tick). While a common progression measurement used for playtraces is time, other metrics can be used: player action frequency, level/-goal completions, and so on. What is important is that these metrics are identified as meaningful by a designer and implemented into a game logging system that allows for post-hoc analysis.

PAS reads a folder from the file system that contains an array of JSON files. The naming of each JSON should be a meaningful identifier corresponding to its playtrace. Examples of good identifiers include time of playtrace or the seed of the generated artifact—something displayed in the interface that will help the user quickly understand which playtrace they are looking at. The format of the JSON is shown in Listing 10.1. Notably, you can have multiple discrete progression measurements and multiple dependent metrics for each progression measurement, allowing the user to switch between different playtrace inspections without reloading data. Discrete progression measurements are represented by their index in their array (i.e., discreteProgressionMeasurements[0] correspond to the first metrics recorded on a playtrace). The dependent metrics must

be numbers for visualization and analysis purposes. This data structure makes PAS game-agnostic—so long as designers define metrics for input (which is likely done as a byproduct of development anyway) PAS can be applied to a dataset.

Once the data are properly formatted, it can be loaded into the system using the PAS interface, and the x-axis and y-axis can be selected using dropdowns. After the playtraces are loaded and metrics are specified, a user can select a toggle to display a point cloud representation of the playtraces they have loaded.

10.3.2 Search Strategy and Output

On the right-hand side of the tool, there is a canvas on which a user can draw an arc. This arc can be any shape, but must pass the vertical line test (no duplicate y values for the same x value), and must only consist of one continuous segment. The user can use the generated point cloud as a guide if desired. The line does not need to reach both ends of the canvas as the line drawn will be transformed to fit within the vertical and horizontal bounds of any given playtrace to which it is being compared. With data loaded and the desired arc drawn, a user can select what search strategy they would like to use: Fréchet Distance, Dynamic Time Warping (DTW), or a weighted combination of both. The Fréchet Distance strategy favors matching based on overall curve trend on a point-by-point basis [6], while DTW cares more about the shape and translated phase of curves and is more forgiving of translations of those shapes (e.g., rising/falling trends) [237]. Both are useful for different use cases—Fréchet Distance might help confirm an overall trend across all playtraces (all metrics are rising/falling in similar places) while DTW can help confirm that all playtraces experienced a similar pattern

```
{
    //System accepts any number of potential x-value
       measurements
    ``discreteProgressionMeasurements" : [
        //Each index in array corresponds with x-value on
           graph
        {
            //Each value in a discrete object can be used as
               a y-value
            //Each value will be present in PAS interface
            ``metric1": number,
            ``metric2": number,
            ``metric3": number,
            ...
        },
        ...
    ],
    ...
}
```

Listing 10.1: An example input JSON schema to be used with PAS. Each object inside of "discreteProgressionMeasurements" should have the same structure. Only numbers can be used for playtrace analysis.

regardless of progression index at which they occurred. A weighted combination of both approaches is also available, allowing for a nuanced mixture of both search strategies. Each strategy scores every playtrace in similarity against the user-drawn curve from 0 to 1. Due to the transformation of the user-drawn curve to match the relative ranges of playtraces, a resampling is performed for both algorithms to standardize scoring. The density of this resampling can be specified using a sensitivity slider below the search method options. The search can be initiated by clicking the button below the search method options.

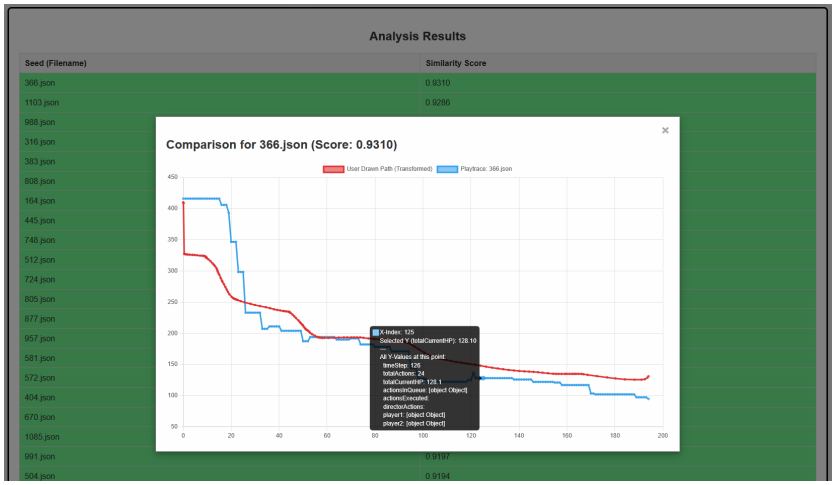

Figure 10.4: A view of the analysis results section with a playtrace inspector open. The similarity score shows how similar the drawn graph is to a given playtrace. Hovering over the playtrace produces all values provided within the input JSON for a given x value.

After performing the search, the user is presented with a ranked list of playtraces, sorted by their similarity to the drawn score (see Fig. 10.4). Each playtrace is labeled by its filename, and its score is shown in its row. Playtraces are colored on a gradient from green to red, with high scores being green and low scores being red. Clicking on an individual playtrace entry shows a graph that overlays the user-drawn arc

with the playtrace, allowing for direct comparison in context. The user can hover over the playtrace, investigating all metrics included in the JSON file for the given discrete progression step.

10.4 Evaluation

To evaluate the efficacy of PAS, we ran an evaluation that sought to confirm three critical design use cases:

1. Can PAS correctly positively score a drawn arc corresponding to the point cloud?
2. Can PAS correctly score a drawn arc that does not follow the point cloud?
3. Can PAS correctly identify a graph with certain features from a large set?

To do this, we used a game testbed with automated playtester functionality to generate 1,000 playtraces [432]. From these playtraces, we generated a point cloud, drew arcs to match each condition, selected appropriate search strategies, and reviewed the results.

Methodology

We used a headless Turn-Based Role-Playing Game simulator named *FighterDDA* for the production of a 1,000 playtrace corpus of data [432]. We selected this system because of its usage of a popular game genre and ability to rapidly generate a large set of playtraces for analysis; collecting a similar volume of human-playtesting would require much more time and is unnecessary to confirm basic system operation.

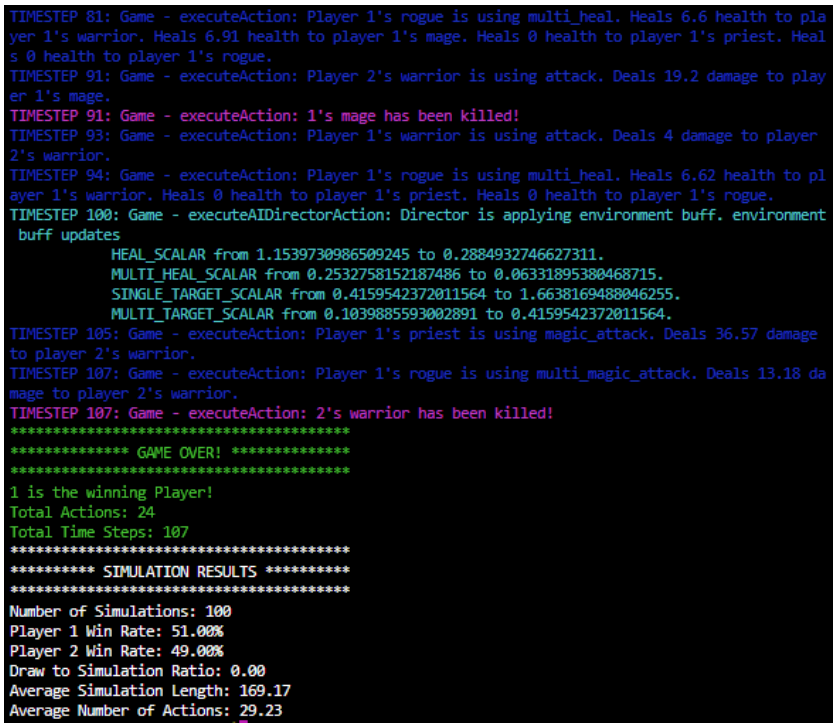

Figure 10.5: Console output from the automated playtesting platform used for evaluation, FighterDDA. This output highlights characters attacking and defeating one another while providing a basic metrics summary. Logging from each simulated fight is saved in JSON format for analysis by PAS.

This system uses utility agents combined with an AI Director to simulate two teams of four characters fighting one another, with the game ending when all characters from a given player have no remaining health points. Characters are able to attack opponents, defend themselves, and heal teammates. The AI director applies environment changes in an attempt to modulate difficulty throughout gameplay. For the purposes of this evaluation, we chose to use the metric of the health of all characters summed together on each game tick. This metric provides us with a view into the general pacing of the game and could help a designer understand if games are ending at appropriate times and if games have some form of back-and-forth style of gameplay. A screenshot of the console output of a run of the data generation testbed is shown in Fig. 10.5.

To test the three cases listed above, we uploaded our corpus of playtraces to the system, selecting game ticks ("timeStepLogs") and total health ("totalCurrentHP") as our x-axis and y-axis, respectively. For case 1, we drew an arc that follows the densest areas of the point cloud and ran analysis using the Fréchet strategy, as we are looking to see if we get matches for the overall trend of our drawn arc. For case 2, we drew an arc that inverts the densest areas of the point cloud, running the same Fréchet search strategy as in case 1. For case 3, we drew a curve that oscillated while matching the overall trend of the point curve, searching for playtraces that held some pattern of rising and falling current health throughout the game. We used a combined approach (both weights set at 0.5) for this case, as we both wanted the overall trend of the curve to be matched, as well as the detection of specific curve features. We set the system sensitivity at 300 for all cases. All drawn curves can be seen in Fig. 10.6.

10.4.1 Results

PAS worked as intended in all three cases listed in section 1. For use case 1 (Q1), we found high scoring of curves with near universal consistency, with scores ranging from 0.7367 to 0.9354. This shows that PAS is capable of matching the overall curve trends that are common within a large set of playtraces. For use case 2 (Q2), we found extremely low scoring of curves with universal consistency, with scores ranging from 0.0000 to 0.3542. This result indicates that PAS is successfully able to identify curves that are highly unlikely to appear within a data set. For use case 3 (Q3), we were able to identify a curve with both overall trend and prominent features close to our drawn curve, as well as to see curves that matched poorly, with scores ranging from 0.6247 to 0.8603. This feature matching demonstrates PAS's ability to identify playtraces with both overall trend and specific features in a playtrace, helping identify exemplary playthroughs of a game. A visual showing the drawn arc and the highest/lowest scoring playtrace for each case is shown in Fig. 10.6.

10.5 Discussion

10.5.1 Mapping Game States to Curves

PAS highlights a common facet of evaluating the quality of game systems and emergent experience—parameters and metrics of the system form curves over a progression, and these curves represent how situations and gameplay unfold. These curves can be viewed as game-system metaphors for dramatic arcs—the rising and falling of these curves over time can help a designer identify the "rising tension" or

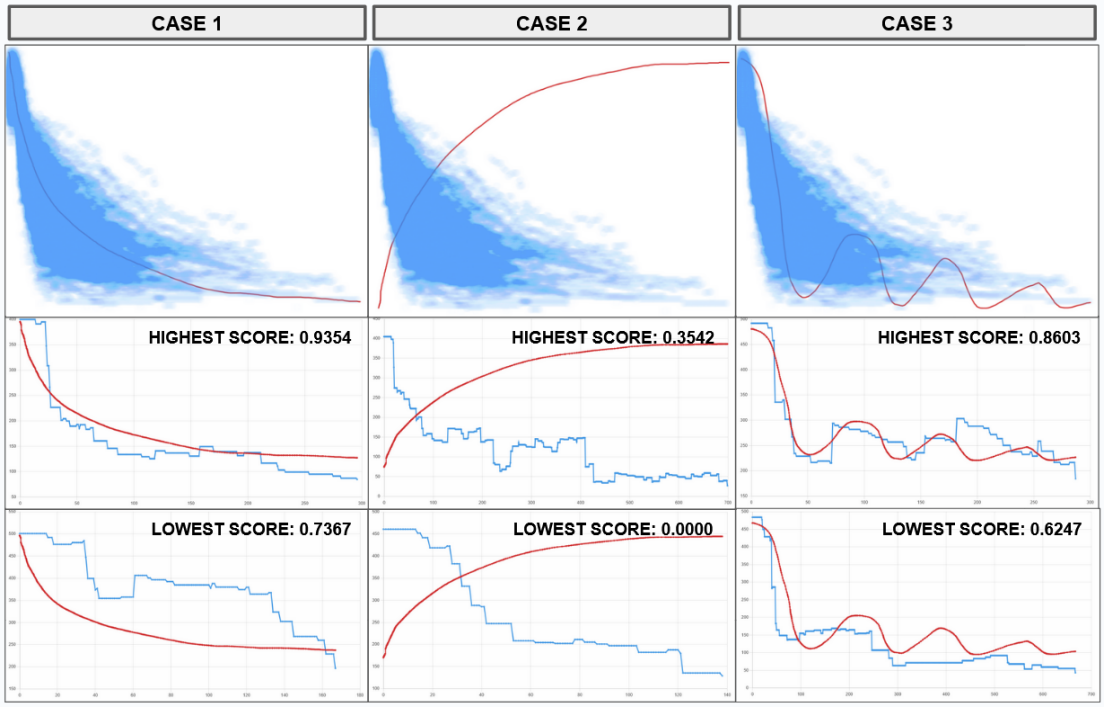

Figure 10.6: Results from the three case studies. Each column shows the designer's drawn graph, the highest matching playtrace, and the lowest matching playtrace. Cases 1 and 2 used Fréchet distance matching (Q1, Q2), while Case 3 used an evenly weighted combination of Fréchet distance and DTW matching strategies (Q3).

other systematic drama over a gameplay session [303].

It is important to note that progression does not just happen on a moment-to-moment, temporal basis. Such approaches might be unnecessarily fine-grained for understanding a game environment. A designer for a game that contains a large volume of small levels, such as a mobile puzzle game, might be more interested in metrics for each level completed over hundreds of levels instead of the minutiae of each individual level itself. The ability for PAS to scale based on the grain of the discrete increment provided allows the tool to be used across a variety of design use cases throughout the game lifecycle—you might investigate the moment-to-moment difficulty of a section of a game, or view how a player changes in skill level over thousands of multiplayer matches.

The ability to see a collection of curves and find important curves also highlights versatility in seeing the macro- and micro-trends of playtrace datasets. In our evaluation, we quickly saw that all the games ended within a similar curve space. This is a useful insight for the platform designer, as it shows that all games are ending appropriately and have a contained space (i.e., an important goal for this style of battle is that battles eventually end and have some level of consistency in how they play out). Meanwhile, it was reassuring to know that the individual character of the curves with a tug-of-war pattern (health oscillating over time) was present, meaning that the games simulated were able to provide experiences that were not overly linear.

10.5.2 Confirming Designer Goals

Designers often use an "experience goal" (i.e., what experience should the player have) to drive their game development process [138]. These experience goals are impossible to know explicitly (as there is no direct access to player experience) and instead must be expressed through qualitative research or system-observable metrics defined through designer-specified player models. Understanding whether a game metric accurately predicts a player experience (and, in some implementations, impacts the game state) correctly is key to meeting these player experience goals. Combining PAS analysis with qualitative data can quickly shed light on whether or not the correct system-metrics have been defined for an audience. For example, a designer might look at PAS and find confirmation for a system curve, but find that players do not qualitatively agree with the data. In such a case, the designer might need to go back to the drawing board to find a better system-derived method to quantify player experience.

When it comes to PCG, Automatic Game Design, or games that have high replayability, it may not be a positive thing to see a consistent set of playtraces. In contrast, the designer of such systems or games might actually be looking for system arc diversity, hoping to see a wide range of curves across playtrace sessions [441]. When combined with an Expressive Range Analysis approach (such as in Shields et al. [438]'s tooling on FighterDDA), understanding the potential design space of system playtraces represents a novel and meaningful strategy to understanding system quality. PAS allows for the visualization of such diversity at a glance with its point cloud, but also allows the designer to see if varied individual curves exist in the gameplay and what specific actions occurred during that playtrace to produce the curve in question. Consistency and diversity can be valid goals for different designers in different use cases, and PAS allows the confirmation of either goal.

10.5.3 Tuning Through Parameter "Sweeping"

PAS is not only useful for looking at a static configuration of game systems—it can also be used for parameter tuning. This can be done by implementing an A/B testing strategy, where different parameter adjustments are provided for each playtest session [202]. With this collection of data, a designer can then specify the output curve they were looking for, identify the playtrace(s) with the highest similarity, and use the parameter settings from those playtraces for future tuning refinement. The scoring from this system could even be implemented as part of a fitness function of an evolutionary loop, selecting games based on the quality of their system arc in addition to other metrics.

10.5.4 Using PAS to find Balancing Drama in Gameplay

The recognition that game systems (whether they are or are not explicitly narrative-oriented) implicitly carry narrative structure, meaning, and storytelling potential is a hallmark of game design [230, 42]. When traditional narrative and game system emergence and progression gracefully work together, games become capable of presenting dramatic arcs in ways that no other form of media can. Namely, players experience dramatic arcs through multimodal gameplay—a fusion of explicit narrative, gameplay feedback, diverse aesthetics, and so on [435]. A player may experience a "rags to riches" arc through a weak character accumulating powerful abilities over time, or a "man in a hole" arc through the gradual reduction of resources and safety in a horror game. Although these arcs can be presented through interactive and dynamic dialogue alone, many games lean on the narrative metaphors that their game systems convey to represent their greater, holistic messages. Using a simple example, the gradual demonic invasion of Earth in *DOOM* is played out through audio recordings and environmental storytelling, as well as the increasing volume, speed, and difficulty of the demon spawns that the main character fights [204]. These dramatic arcs, the result of the combination of traditional storytelling and system-driven storytelling, help define the overall sense of progression in a game.

However, the emergent properties of game systems along with the lack of generic translations between game system-driven narrative and traditional storytelling make predicting the progression of a system's dramatic arc difficult. A system's embodiment of difficulty, balance, tension, or drama either has to be statically defined by

a designer or built into procedural systems that might have a wide range of potential dramatic system arcs that could be produced. As such, providing tools that help a designer directly understand the character of a game system's progression through the lens of dramatic arcs could prove invaluable for the careful tuning of a multimodal, interactive game narrative.

We can conceptualize the systematic narrative arc of a game or interactive digital narrative through the lens of playtraces, defined by a relevant designer-identified metric and a measurement of experiential progression. Using PAS to view playtraces, we can leverage an iterative process to concretely evaluate if 1) there is a system-observable metric that defines game narrative progression, 2) if the defined metric's changes over a progression form a set of meaningful dramatic arcs as desired by the designer, and 3) if those arcs correspond with the qualitative experiences of the players who interact with the system. We call this iterative design process a "Dramatic Arc Search". A visual of this iterative loop is shown in Fig. 10.7.

Finding Drama Metrics of Game Systems

Dramatic arcs, in the context of game systems and playtraces, are defined on their y-axis by a (usually) abstracted measurement of what a player is experiencing at any given moment. There is a nearly infinite variety of meaningful dramatic metrics—it is dependent on the designer, their experiential goals, and the specific game context to identify what computed metric represents the game system's dramatic arc. For the process proposed in this work, the metric must be numeric in order to be plotted and analyzed in the context of playtraces.

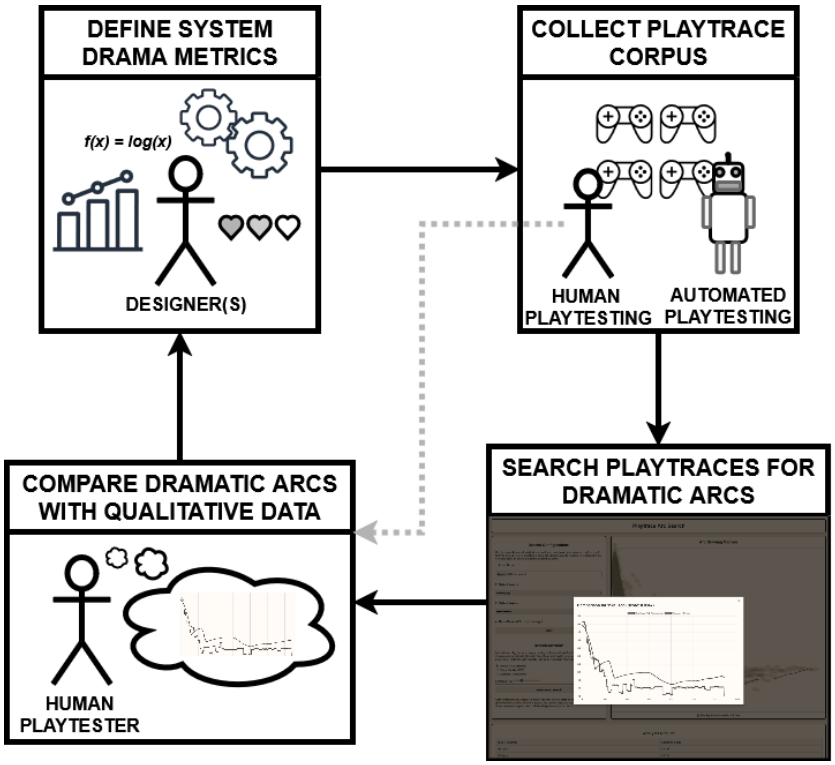

Figure 10.7: The iterative design loop for understanding how dramatic arcs are being presented within an interactive system.

Previous examples in this work highlight more complex versions of this, with opaque definitions such as "emotional intensity". That being said, simpler y-axis definitions can still yield productive designer insights—if we are designing turn-based RPG battles, we might want decisions such as attacking, target selection, and healing to generate a back-and-forth conflict that gradually declines until one party's health has been depleted. As such, a simple measurement that might provide insight into dramatic arcs for this system is the overall health measurement of all characters. We would quickly know if the system is not producing dramatic arcs correctly for battles if overall health gradually rose instead of falling over time, and we would know if games are boring if all playtraces followed too similar of a trend within overall performance.

The x-axis of playtraces represents the progression of the game, the increments in which the arc metric is measured by the system for future reporting. The progression can be tied similarly to a wide range of simple and complex measurements. Obvious examples include time elapsed or game ticks, but designers might use custom progressions such as critical checkpoints, overall game progress, or number of player actions. The x-axis represents the narrative progression in traditional linear dramatic arc plots; what story beats have elapsed at a given point in the narrative. As such, our playtrace definition of the x-axis should similarly be thought of as systematic beats—inflection points where the system can alter the arc the player experiences. Continuing our turn-based RPG example, we can take the simple road once again and define game ticks as our x-axis, as they represent moments where a character can prepare or execute an action that impacts the overall health of characters in the game.

With the x- and y-axes selected for our narrative arc, we can construct play-

traces that capture the arc throughout the gameplay. These playtraces are recorded during human or automated playtesting, capturing the range of potential arcs that occur for a given system. For the illustration of our process, we will use overall player health over game ticks in the FighterDDA environment over 1,000 playtraces to evaluate what dramatic arcs exist in the platform's battles. If this playtesting is done with human players, it is crucial to also collect qualitative data about their play experience so that the selected x- and y-metrics and their corresponding playtrace arcs can be validated as meaningful indicators of the actual play experience.

Using PAS for Systems Drama

With formatted x- and y-axis data that correspond to a designer's conception of how a system contributes to a dramatic arc, it is now possible to log gameplay data into a playtrace and represent it as a graph. While individual playtrace case studies are useful in their own regard, it is difficult to understand the true dramatic potential from a single or limited number of playtrace examples. The collection of a large corpus of playtraces allows for a more complete understanding of playtrace diversity, giving a designer insight into a system's dramatic potential.

PAS reads a large set of playtraces and provides tooling to 1) see the overall distribution of playtrace data through a point cloud and 2) search for designer-defined dramatic arcs within the playtrace dataset. A screenshot of PAS with descriptions of its basic usage can be seen in Fig. 10.8. These two features enable a designer to quickly see if a desired (or undesired) arc is possible for a game system given a playtrace dataset of sufficient size. Fig. 10.9 shows how a designer might use the curves specified by

(Reagan et al., 2016)'s analysis to see if such patterns exist in their data [383]. It is quickly apparent that the "rags-to-riches" arc will never occur in this dataset, which would likely be preferred by the designer (as infinitely increasing overall health would indicate RPG battles that never end). On the other hand, an overall gradual decline of health consistently exists, with some presence of curve variety in different sections of the playtraces.

A designer can, after seeing the traces in aggregate, sift through all produced system arcs by drawing a curve that they are curious about in their system and performing a search for similar curves. For example, a designer might want to see an oscillation of the "Oedipus" (fall-rise-fall) style throughout the overall trend of downward curves shown in the point cloud. They can look for this pattern by drawing such an oscillating curve within the bounds of the point cloud and seeing if any playtraces meaningfully match the drawn curve. Dramatic arcs can happen in local or global settings, so defining when and at what scale curves occur in a playtrace arc can be understood by the designer by carefully defining their search curve accordingly.

Combining Arc Visualization with Qualitative Data

Once playtraces have been collected and a designer has inspected the resulting system dramatic arcs using PAS, it is important to combine the observations of the system's progression with qualitative player feedback. The danger of system-defined metrics for narrative is that they cannot fully encapsulate player perspectives, and it is quite possible that a designer's educated guess at what system metrics reflect overall tension and drama in players is vastly different from actual player sentiment. Thus,

Figure 10.8: A screenshot of the Playtrace Arc Search tool. Users can upload a playtrace dataset, select x- and y-axis definitions to analyze, and draw a curve that they'd like to search for within their trace corpus. Similar curves are matched based on either Fréchet Distance or through Dynamic Time Warping. Results are presented after clicking the "Perform Arc Search" button, with matches scored and colored according to fit. Optionally, users can enable a point cloud to view the global space of playtraces as well as inspect individual playtraces as compared to their drawn arc (see Fig. 10.9 for an example of point cloud and trace inspection functionality).

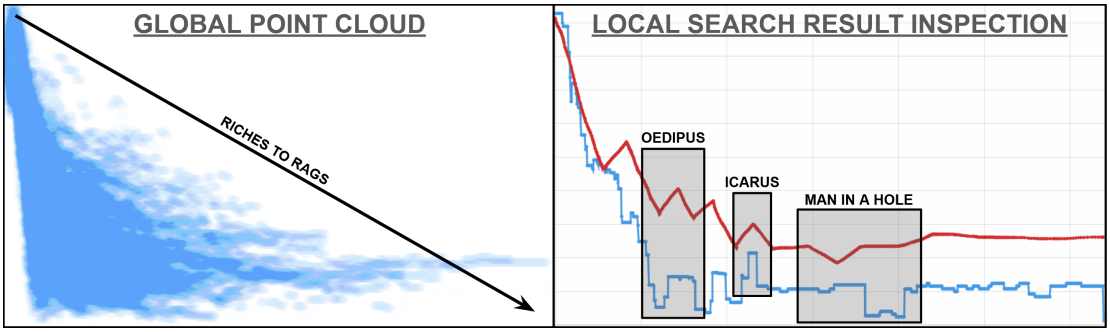

Figure 10.9: Global point cloud (left) and individual dramatic arc search results (right) produced by PAS with annotations highlighting dramatic arc structures. The global point cloud shows every data point within a playtrace corpus, allowing a designer to see density of playtraces over time. A cursory glance reveals all playtraces roughly follow a "fall" dramatic arc. The individual inspection shows an individual playtrace with a high search score, with the designer-defined search arc in red and the specific playtrace in blue. Multiple local dramatic arcs can be identified throughout the playtrace. The results shown in these images are produced by running PAS against data produced by FighterDDA.

after a designer has validated that their system is bearing out the arcs they desire in their playtrace set, they need to investigate if the players are also experiencing the game as desired and indicated by the playtrace set. If there is dissonance between the qualitative player experience and the quantitative measurements of the playtrace arcs, it becomes necessary to return to the first step and find more relevant and useful system metrics to use in order to evaluate a system's dramatic character.

In the case of FighterDDA and the search of "Oedipus" curves, we would want to see if players experience an engaging battle due to the oscillation of health over the course of a battle. If they do not, it would become necessary to find (or calculate) another metric for further playtrace analysis.

10.6 Limitations

We believe that PAS has the potential to be a valuable tool for designers to rapidly iterate and evaluate their games based on systematic playtrace curves that might be difficult to parse or observe without it. However, there are clear limitations that could be addressed in future work. For one, a trial with game designers across a varied selection of games would greatly bolster the tool's real-world usefulness and ensure that it has the right usability and visualization approaches to aid in game design. Our evaluation confirms basic system functionality, but would benefit from being applied to a wider range of playtrace corpora, especially human-generated playtraces. Applying this form of evaluation to more complicated and a wider diversity of system measurements would also be useful. For example, it would be interesting to see how PAS could be applied directly to narrative spaces to see if it could directly capture the dramatic arcs that it uses as a metaphor for system analysis. Integrating the arc-matching process into generative search strategies (i.e., into fitness functions) also represents an interesting design potential in looking for system quality when generating levels or games. Implementing a database adapter (so that data can be queried from a database rather than saving JSON files to disk) would also reduce requirements to improve/alter existing data logging systems in games and improve PAS compatibility overall. Finally, common arc patterns (such as the narrative arcs described in Reagan et al. [383]) could be provided to users as an alternative to drawing, effectively building on the work of Leong et al. [271]'s approach to mapping storysifting to Reagan et al. [383]'s definitions.

10.7 Conclusion

Playtrace Arc Search is a tool that allows designers to evaluate a large corpus of playtraces through a visualization tool and search for specific systematic "arcs" within their dataset. It provides visuals for the entire dataset as well as inspection of individual playtraces and data points alongside a scoring system that shows the similarity of a given dataset to a user-drawn arc. The tool contains multiple search strategies and proved to be successful in searching through a corpus to find examples of relevant curves drawn by a user. The combination of generic input, the ability to rapidly iterate through arc searches and strategies, and the range of visualization output indicates that PAS can be a useful tool for designers who wish to understand the quality of their game systems over time through playtesting data.

Additionally, The iterative process presented under the discussion revolving around system drama provides a method to understand system-driven dramatic arcs of games through playtrace analysis, drawing a direct conceptual and visual metaphor from dramatic arcs in traditional narrative. The advantage of this process is that it allows designers to more concretely understand how system progressions and player interactions over time can create dramatic arcs at different critical points of their game. Such a style of analysis and tuning can, due to its system-agnostic character, be applied to any dynamic game system to look for desired peaks and valleys over a gameplay session. The design process presented can help confirm that a system consistently delivers increases in intensity at predefined points in the game, or that a system produces a wide variety of arcs depending on player choice.

Both of these outcomes are valuable depending on designer goals, as one guarantees a specific narrative plays out while another guarantees that an interactive system has a wide expressive range. Connecting PAS analysis with qualitative data also helps designers understand whether their quantitative metrics actually align with the experiential goals they have for their system. The process does have limitations—the designer must have expert knowledge about their game system, the ability to identify and compute key system metrics, and the knowledge to be able to understand their desired arc outcomes.

Between the last three chapters, I have presented technical work that covers proceduralizing static balancing, making procedural dynamic balancing cater to different user personas, and how to evaluate balancing systems for expressive texture and diversity. Together, this technical work demonstrates the framework and taxonomy put forward in Part II of this dissertation, while providing new technical approaches to building out procedural systems with players in mind.

Part IV

What's *Next* for Game Balance?

Chapter 11

Future Opportunities

In this dissertation, I have presented balance as a multidimensional term with a myriad of potential implementation strategies, both manual and procedural. I have argued that it takes a finely-focused design approach, segmenting audience, developing goals and hypotheses, implementing a bespoke method, testing, and doing it all over again as necessary. I have presented applications of this design framework, as well as suggested evaluation approaches that can save time and focus insights on the game balancing process.

My research journey was a rise-fall-rise narrative—I managed to build a procedural balancing system, then discovered the failures of not being tactical enough with my evaluation and design process. I used that failure to rework my understanding of the concept of balance, building out a taxonomy and design process to better handle the nuance and texture of game balancing. This enabled me to build a new system that included more specific personas. The realization of these different personas and the underlying foundation of fairness in balance made me realize the potential of analyzing the

progression of balance throughout a game, motivating me to develop expressive analysis tooling to better frame how we balance games.

All of that said, I have necessarily implemented isolated, segmented sections of balance and procedural implementation. This is to prove out the process and constrain the potential output of systems to better suit experimental reporting. As such, there is a *large* space for future research, especially as it relates to permuting different types of balance and addressing less-focused on areas of game balancing. I have found the following ideas fascinating and worth more research, but alas, it would take another doctorate (or two) to cover them. As such, I present some starting points in the hope that I inspire future researchers to dive deeper into player-centric, procedural balancing.

11.1 Hybrid Architectures

In Chapter 6, I presented the distinction between static and dynamic balancing approaches, framing dynamic balancing as balancing that occurs during the runtime of the game and static balancing as balancing that occurs during design time. Dynamic balancing might be an AI director tuning difficulty in response to player performance, or even just a base reduction in obstacles for players who are struggling when a level reloads. Static balancing is the tweaking of parameters that designers do before the game starts to maximize their goals. Does this jump height make the level too hard, too easy, or address player needs in general? In my technical exercises, BrawlerAGD was framed *BrawlerAGD* as a system that seeks to statically tune games, even with a sophisticated playtesting and scoring loop. In *FighterDDA*, I present a dynamic system

that is reacting to player input and tuning in response.

While these separations are useful conceptual separations and can help us focus on specific problems, the reality is that one often necessitates the other, and that real-world balancing likely combines the two for a final product. I found that *FighterDDA* was emblematic of this point. While I was implementing the director system, I found that the director itself was a parameter-driven agent that required tuning to behave properly (the frequency of interventions, thresholds for intervention, etc.). Although I did not implement a procedural balancing system to find these parameters (though in retrospect I wish I had given the difficulty of finding the right combinations), it could have greatly benefited from a tuning strategy as presented by *BrawlerAGD*. Static balancing also benefits from the tooling and systems produced by dynamic systems—developing fitness criteria, collecting data from automated evaluations, and seeing the interaction between the dynamic balancing system and the static tuning of the rest of the game. All of these systems developed in the process of *BrawlerAGD* tuning system prove this thought out, alongside presenting opportunities for dynamic systems if desired by the designer.

The point is that there is work to be done in expanding how we view hybrid static/dynamic procedural balancing architectures, and how the integration of these architectures could provide higher quality game artifacts.

11.2 Aesthetic Procedural Balancing

The work done throughout my degree generally focused on game system tuning over aesthetic tuning (aside from a foray into believability in *FighterDDA*). This was primarily due to time, skill, and scoping constraints on my part. This should not be taken that I think that looking into how the aesthetics of a game impact balance is not a worthy endeavor. Quite the opposite—I see a particularly important and underexplored need to push forward in this space.

Interviews with designers shed light on this facet—the previously brought up example of using lighting to balance a level in *Halo*. Similarly, tweaking the sound of a gun in *Wolfenstein: Enemy Territory* changed the minds of players from thinking a gun was comparatively underpowered to a mechanical equivalent to thinking it was properly balanced, without any system, damage or performance changes [366]. The progression of aesthetics reflect a sense of player pacing, with the ramping visual and sound effects of weapons in the *DOOM* games reflecting the power fantasy of becoming more and more deadly to demons throughout the game. The tuning of sensory elements (hearkening back to qualia-driven design) presents a procedural balancing opportunity—how do we find the right configuration of aesthetic elements such that we evoke a feeling of balance? Going a step further, the telegraphing of information also in part determines a game's balance. When asked about tuning randomizer balance, Stella Mazeika said "You're going around in Ocarina of Time and in a place where you would visually see a Heart Piece that you would be able to get to. Instead it's the Light Arrows. You can visually see that check there and get that information. That's actually one of the really

interesting tools and meta shifts that comes into play: how do you control that sort of flow of information?". Similarly, the performance and difficulty of speedruns depend on the knowledge of shortcuts, glitches, and so on, as discovered by a combination of the community, available design documentation, and developer commentary.

It also ties into procedural content generation and the balancing applied to a breadth of generated content. I performed an initial test on *BrawlerAGD* where I attempted to procedurally create sound effects according to the properties of parameters that defined the game systems. While it showed some promise, my lack of ability to quickly implement sound effects meant the effort took too long to apply system-wide, and I couldn't get a user-facing test going. If I had, I can see using PAS to investigate how different aesthetic configurations encouraged different playstyles or game pacing. However, a limitation of this experimental environment is the increasing need for human playtesting to evaluate success. If systems needed human playtesters to confirm, more holistic, aesthetic decisions would need it even more so. That being said, I can see aesthetic balancing as a major place for future work in the field.

11.3 Procedural Balancing for Procedural Games

While there is some debate on the time/cost trade off between manual and procedural balancing systems (e.g., Ian Schrieber describes balancing as a "solved problem" due to the existence of mathematical models, playtest loops, and spreadsheet tooling), procedural balancing seems to be a necessary step for AGD and PCG systems. If one cannot possibly test every emergent output of these systems, it is useful and beneficial

to the player to have some conceptualization on whether a given generated artifact is balanced or not. *BrawlerAGD* makes this explicit (the point being to generate games with balance as a selection criterion), while *FighterDDA* also implicitly represents this by demonstrating how diverse playtraces can be interpreted as balanced and expressive.

A pattern seen in both PCG and AGD is that of evaluation strategies. Some of these are atomic and tie to basic game characteristics (game length, player actions), while others are computed and speak to a game's completability or expressivity (e.g. beats). I suggest here that game balance could be another such evaluation property. It can be used during generation to select for better game examples, or post-hoc to understand what generated artifacts performed the best in regards to balance. Combined with a keen design eye for audience and goal, solving for stale or uninteresting games might be a matter of watching playtrace lines for interesting inflection points or dramatic arcs and tuning accordingly to meet a drawn tweening curve.

11.4 Human-In-The-Loop Balancing at Scale

Human-in-the-loop systems involve generative strategies that include a player (or designer) in the creation of a game artifact [270]. Instead of an automated fitness function, the humans themselves become judges of the artifact's quality, helping to provide guidance for a computational system to make decisions on what design steps are needed next. *Galactic Arms Race* is a touchstone for this type of game evolution, where a player character's weapons gain new characteristics based on how the game was played [181].

The relative frailty of automated evaluation systems due to the need to scope the function to a persona's requirements and a reliance on an automated playtester that may or may not act human means that replacing these elements entirely with human decision making could make the quality of a search strategy increase dramatically. Although certainly slower than automated playtesting, this trade-off with validity in evaluation might prove to be a better strategy than automating larger parts of the system. Again, if we automate enough such that we save even a few percentage points of effort, we have won a major battle.

Beyond search quality and automation, a unique design space opens up that appears to be underexplored. That is, game environments that balance themselves organically as a result of player interaction instead of explicit designer tuning. As the game is played and player performance and sentiment are recorded, a system could select tuning strategies and deploy them across other players, repeating this process in a genetic or learning loop that moves around the game tuning over time. What this would result in is unclear but tantalizing—would the game over-optimize itself to the most strategic of players? Would influence from different personas result in different design spaces? Could new patterns of emergence occur from the accumulated changes in the system? If we pair such strategies with personas or psychographics, could we tune for specific audiences? All of these are worthwhile questions in both academic research and the future of game design writ large.

11.5 Personas and Psychographics

I spent a decent amount of this work focused on how important the desired player persona group is to understanding how balance will be interpreted upon game release. That being said, I tended to lean on more traditional, designer-dependent methods of detecting and developing for different psychographics of players. While effective, there is room for error in this approach, as well as the elimination of runtime identification of these groups for more bespoke tuning without studio interventions.

Luckily for us, player modeling and psychographics is a well-researched field that is continuing to expand as time goes on. [445] work, for example, focuses on using clustering techniques and generated player models to understand player actions and generate content centered around that player model. Work has been done using existing or readily available telementry to develop player and group profiles, using existing logs over a long period of time in games like *Everquest II* [439, 100]. The point is—there is promise in the ability to identify player personas and then subsequently tie them into our balancing strategies. Doing so could help create intensely personalized experiences or encourage the emergence of new interactions in a wider design space of procedural game systems.

11.6 Generative AI and Game Balance

I want to scope the term "generative AI" here, as I am not using it to refer to the PCG or AGD methods espoused earlier in this work or more generally in the technical games research community. Instead, I am using it as a catch-all for LLM,

Diffusion, and Generative Adversarial Networks that require massive amounts of compute and data to produce output. There are a litany of issues with LLMs outside of game balancing: they are an ecological disaster [161, 149, 443], blatantly plagiarize and violate copyright from creatives [274, 392], guide creative ideas into convergent buckets [12], damage pedagogical institutions [223, 504, 105, 179], encourage mental health degradation [238], and eliminate entry-level roles while applying significant workload to mid- and senior-level roles [314, 160]. These are well documented elsewhere, though I wanted to put it front of mind when considering LLM usage in game balance (or any game design task).

Putting these concerns aside and focusing on the affordances of these tools, I am somewhat skeptical of opportunities to integrate generative AI into game balancing processes. We have hammered home two key elements of game balance: it is emergent and dependent on player perception, and it generally involves the manipulation of numeric or enumerated properties of game systems. These are two things that generative AI currently struggles with greatly and it is unclear if there is a capacity to solve such issues. Without situatedness in the way that a human is (i.e., their reasons for perceiving preferences as determined biologically and socially), the opinions of generative AI systems are currently not able to perceive the impacts in differences in tuning, especially with the subtlety that often comes with such a process. Because they aren't deterministic, they don't carry the same understandability that a more traditional fitness or evaluation function might have, further reducing their utility. Generative AI tends to fail when faced with raw numeric manipulation, especially when it is inherently tied to the emergent phenomena detailed above.

This is not to mention the enormous cost of building and training these models. Say, in an attempt to mitigate the problems above, an LLM was created solely off of the game, game data, and inspiring media around the game. Even if one were to find success in this approach, the hardware, time, and tuning resources would be so large that it would beg the question: why not just use the cheaper, more efficient, and proven process of playtesting loops or established procedural approaches? Even using a smaller-scale ML approach would be vastly cheaper and better fit a bespoke game environment than a more generic generative AI system.

All of that said, I am not oblivious to some of the potential use cases of these systems. I admit to using them in paper editing and the building of visualizations (after I have data formats and procedures well prepared), as I find that these accelerate my work where I normally get bogged down finagling libraries or control+f regex. I am even somewhat hopeful for the growth of local and small language models, hopefully mitigating ecological and copyright issues. I can also not deny the rapid progress and pace of these systems in the last half-decade. The capabilities I malign above might be solved in the next few years, enabling the application of such models during game balancing. Perhaps game states can be converted into LLM-interpretable documents, or the agents produced by LLMs will be able to occupy a more player-centric approach to balancing than the existing combination of designer intent and algorithm tuning. The space for future research exists, and I welcome development in this space, especially if ecological and intellectual property concerns are addressed in parallel.

11.7 Identifying Destructive Deployments of Balance

There is a dark side to game balancing, personalization, and even PCG writ large that I have hinted at but wanted to explicitly address before the end of this work. Namely, if balance is meant to target optimal challenge and engagement, and it is possible to tune this engagement at runtime, we run the risk of falling into the same dark-design patterns often leveraged by various advertising and social media platforms. This, combined with questionable monetization models, means that the ethics of applying balance design can become *im*balanced when it comes to building games that encourage dramatic, interesting experiences instead of games that seek to trap and exploit our attention.

There is no shortage of sensational news reports and more reasonable academic literature on the risks of video games dominating someone's life or wallet, especially in relation to children [501, 262, 289]. Some of this is certainly overblown, and as someone who can easily sink hundreds of hours into a *Pokemon* game and is a lover of *Magic: the Gathering*, I question the bright-lines drawn between engagement and attentional abuse. I think balance can, for better or worse, be used for both ends. As such, I think a key area of future research is fleshing out which styles of balance are more exploitive versus which enrich a gameplay experience.

I believe that ethical balance design can be moderated by keeping a game designer's motives around play in mind, tuning balance around the artistic and expressive affordances of games instead of a distanced perspective on maximizing time-on-screen or revenue. In the same way that writers might prefer to maintain authorial intent and

creative control, game designers would do well to maintain their creative control that focuses on the message and meaning they want to convey. The beauty of *Elden Ring*'s balance is not in that it appeals to everyone, to keep everyone hooked, but in that its difficulty progression puts the player in the character's shoes—failure, learning, mastery, victory over seemingly insurmountable odds. We play the downloadable content for *The Legend of Zelda: Breath of the Wild* not because we felt we needed it to get to the next level, but because our accumulated skills and toolkit begged for application in a comparably challenging context. In both of these cases, the designer's intent on perilous adventure shines through to the player.

Chapter 12

Conclusion

"When the player fails, they want to slap their own foreheads, not the designers'."

Ian Schrieber, Author of *Game Balance*, designer for multiple studios, Educator

You may have asked during the introduction—what drew me to the competitive side of *Super Smash Bros. Melee*? What was it about my persona that pushed me towards tournaments and away from the neighborhood? The answer has many facets, but the one that drew me in the most was the idea that the game had such a polished set of rules and interactions that its gameplay was productive beyond the vision of the designers. With a robust physics system and beautiful character tuning, unexpected interactions of the game environment itself led to new ways to play, precise forms of movement, mind games, and so on. In my doctorate, I realized that a precisely tuned

system isn't just about player engagement—it is about widening the design space of a game itself so that more and more modes of the game cut deeper. Moving to *Brawl*, slowing down movement and disabling some of these unexpected interactions greatly reduced the productivity of the game engine in terms of gameplay. Simply by tuning parameters!

Finding the right tuning and balance of a game means getting the outcomes you expect and getting outcomes you don't expect (that somehow still carry quality). It means players feel like they always have a shot, and if they don't, that they one day will with enough practice and patience. It means that different groups of players can find the games that fit them best, or they can find different modes of play personalized to their passions. Children can enjoy the main storyline of *Pokemon* with little failure, and adults can dive into hidden systems and sophisticated combos in competitive play to meet their needs in complexity and challenge. Collectors can seek exceedingly rare "shiny" creatures, speedrunners can over-optimize the main quest, randomizer and nuzlocke runners can create community-driven styles of play customized for their desires. The scale of each of these communities speaks to a brilliance of balance in *Pokemon* not always appreciated—its framework for play enables a wide range of people to engage with the environment with a great scale of depth. All of these modes depend on balance—stats and moves, appearance rates of creatures, probabilities of critical hits, and status conditions. Even with the formula relatively unchanged over the years, *Game Freak* has done well to tweak these facets under the hood to keep the game balanced for nearly all of these players, even if their playstyle isn't officially supported by the game or studio.

In terms of explicit contributions, I have done the following:

1. Placed game balance as emergent and in terms of the experience of fairness
2. Defined the variety of definitions used to scope and frame balance
3. Provided a framework to apply balance regardless of game context
4. Gave a catalog of potential targets and methods for game balancing
5. Suggested evaluation strategies for game balance
6. Tested static balancing in a fighting-game, AGD setting
7. Investigated dynamic balancing in an RPG, AI-Director setting
8. Released a generic tooling for quickly investigating balance through corpora of playtrace data
9. Suggested future directions for game balance research

If you only come away with one thing, I want it to be that game balance is the act of game design as it is viewed through fairness. It involves how we tell stories of fairness through our gameplay—how we feel outmatched, dominant, and neck-and-neck throughout the course of a game. Game balance forms a narrative in players through the way that they traverse an ever-shifting landscape of fairness judgments. The design of balance is subtle and complex and often lives in the vasculature of the near-invisible systems that form the beating heart of video games. My work gives designers and researchers ways to navigate those systems, to make the invisible parameters visible, to help expedite and sharpen adjustments to those systems. All of the work presented here

aims to acknowledge the rich aesthetic power in conceptualizing games as a tug-of-war game of fairness while adding levers for designers to better engage with the systems that inform said fairness.

If you have made it this far, you have gained an appreciation for the variety of players and the diversity of styles of balance. You have learned a heuristic for focusing on balance in your game or research, and got a briefing on common methods used for the practice. You have learned evaluation strategies and have seen all of the above put into practice. You have learned that balancing is not just tweaking numbers for win-rates, but that it can be deeply tied to a game's expressivity, design and play space, and the ways it can be played by a player base. You have seen areas of experimentation that hold promise for future research in the space and been cautioned against misuse of balancing or balancing techniques. You have put up with a high volume of balance puns, and I have somehow gotten to the end without using the obvious "Thanos" quote.

Thank you for joining me on this journey. I hope with this tome in hand that we can use balance as a lens to create games that help us to form a conversation with our own senses of fairness and challenge us to grow, play, and design with ease.

Appendix A

Game Designer Interviews

There is a well-known and considerable gap between academic experimentation in game design and industry practices. This is likely for two reasons: first, studios and publishers view their intellectual property and the tools used to create it as their asset; releasing it into the would might constitute a loss of their "special sauce" and a corresponding loss of revenue. Second, academics tend to focus on edge-cases and emergent technologies. While interesting and useful in the long-term, these experiments are often time-consuming and do not always guarantee success. Such an outcome would be disasterous for a game release where money and employee effort is on the line.

As such, towards the end of my degree, I wanted to start collecting stories of designers in the field across randomizer, Indie, and AA/AAA studios. I wanted to see the realities of game balancing and how they influence design decisions that studios and publishers care about. While this research is ongoing, at the time of writing I had collected 12 interviews with designers. Quotes from these designers are peppered throughout this work, and I plan on submitting a consolidation of all interviews as a

publication in late 2026.

A.1 Methodology

Interviewees were recruited through a snowball method from my personal network across LinkedIn, Discord, and Bluesky. Participants were asked to be recorded (so that I could have a transcript) over a zoom call for a minimum of 30 minutes. During those 30 minutes, I conducted a semi-structured interview oriented on their perspectives on game balancing. Questions I included in every interview are:

1. What is your working definition of game balance?
2. Can you share an anecdote surrounding game balance in your career?
3. What are prominent examples of good/bad balancing you've noticed?

Interviews then naturally flowed from these topics, diving into the details of their balancing process. Participants had the option of taking any level of anonymity they chose, and could redact any information they did not want on the record.

A.2 Participants

Name	**Indie/AA/AAA/Modding**
Tyler McCarthy	Indie
Melody Ju	AA
Ian Schrieber	Indie
Oscar Gonzales	Indie
Martin Pichlmair	Indie
Charlene Putney	Indie
Ravyn Hardcastle	Indie, Modding
Brie Chin-Deyerle	AAA
Dominick Reba	Modding
Stella Mazeika	Modding
Ross Mawhorter	Modding
Michael John	AAA

Appendix B

Supplementary Links

Below are links to resources mentioned throughout this work to supplementary resources such as code, datasets, and archival forum discussions.

B.1 Reddit Discussions on Balance

The following are reddit links for posts regarding balancing of *Animal Crossing* and *Super Smash Bros.* These posts can be entered into an archival service (e.g. WayBack Machine) with the date August 29th, 2025 to find their fetched context. This date is used because it is before Reddit stopped tracking many community statistics.

Super Smash Bros. Links:

1. `https://www.reddit.com/r/SSBM/comments/1mq8zrw/is_it_time_to_ban_unfrozen_pokemon_stadium/`
2. `https://www.reddit.com/r/SSBM/comments/72j8mn/what_do_you_think_is_are_the_most_balanced_or_even/`

3. https://www.reddit.com/r/SSBM/comments/wmo3op/i_analyzed_almost_25_million_slippi_unranked/

4. https://www.reddit.com/r/SSBM/comments/1m7p1hf/what_are_some_of_the_major_nerfsbuffs_youd_add_to/

Animal Crossing Links:

1. https://www.reddit.com/r/AnimalCrossing/comments/gfzizb/villager_personality_balance/

2. https://www.reddit.com/r/AnimalCrossing/comments/fyt9xr/sound_balance_in_nh/

3. https://www.reddit.com/r/AnimalCrossing/comments/1g13bb6/grinding_in_ac_is_too_unbalanced/

4. https://www.reddit.com/r/AnimalCrossing/comments/fo7y7y/a_poll_who_thinks_the_durability_is_balanced/

B.2 Literature Review Data

Below are links that include the data used for the literature review in chapters 4 and 5.

1. Search Strategy: https://docs.google.com/spreadsheets/d/e/2PACX-1vSJmEd19IilHXIV7lrUViIhVzqInNk_ML-Bk5vmuXM2-h9IxIt45wUCz-wlBmvWOBPIinxQVjEppFkV/pubhtml

2. Taxonomy Code Processing: https://docs.google.com/spreadsheets/d/e/2PACX-1vQPjy7Qjva9loMDLtRyBdp1JLk42eW7szLU3MAQidOpWZO0uTIWPdhY9u4WBoCKm5-NjjlaWBc5XPGs/pubhtml

3. Framework Code Processing: https://docs.google.com/spreadsheets/d/e/2PACX-1vSnYFBB8LkITdxMuZo4zHJVYlAXKUPk-Eki-PZComn9G0PQ-Bh9gi8tqtIL6VlqtyyGC_b6E7Ukk90u/pubhtml

B.3 Code Repositories for Technical Work

All technical work presented in this dissertation is available on my Github. Links to specific projects are:

1. BrawlerAGD: https://github.com/smshields/BrawlerAGD

2. FighterDDA: https://github.com/smshields/FighterDDA

3. Playtrace Arc Search: https://github.com/smshields/ArcSearch

Appendix C

Genre and Platform in Balance Research

During the literature review exercises performed in chapters 4 and 5, we also accounted for the list of games and genres used in the balance literature. The resulting data showed a wide range of game genres (figure C.1) and platforms (figure C.2 used for analysis. This seems to be because the academic literature reviewed largely focuses on creating an approach to automated or analytics-driven balancing efforts and as such are implemented and understood through the context of a specific game, genre, or platform. More theoretical efforts to categorize and reflect upon balance lacked this specificity, opting for a top-down approach to evaluating the concept [33, 342, 157, 26]. Some publications focused on developing their own game in a given genre to investigate balance. Examples include [13, 217, 481, 268, 7, 500, 36, 31, 322, 232, 201, 291, 403, 502]. Others leaned on existing, published games to do their investigations, including

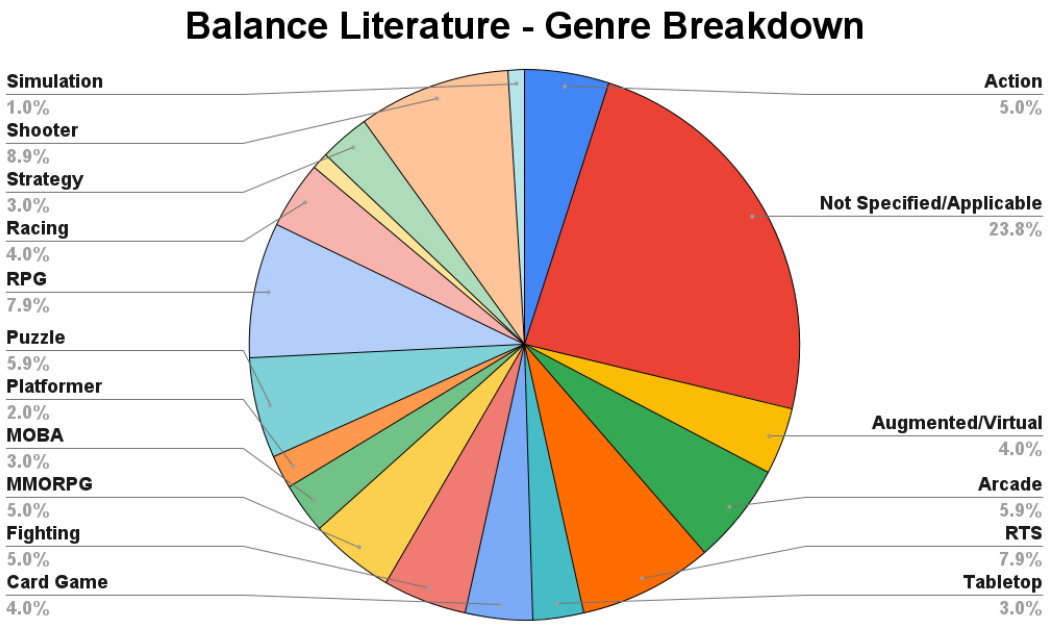

Figure C.1: Pie chart listing the genres that publications used to focus their investigations of game balance.

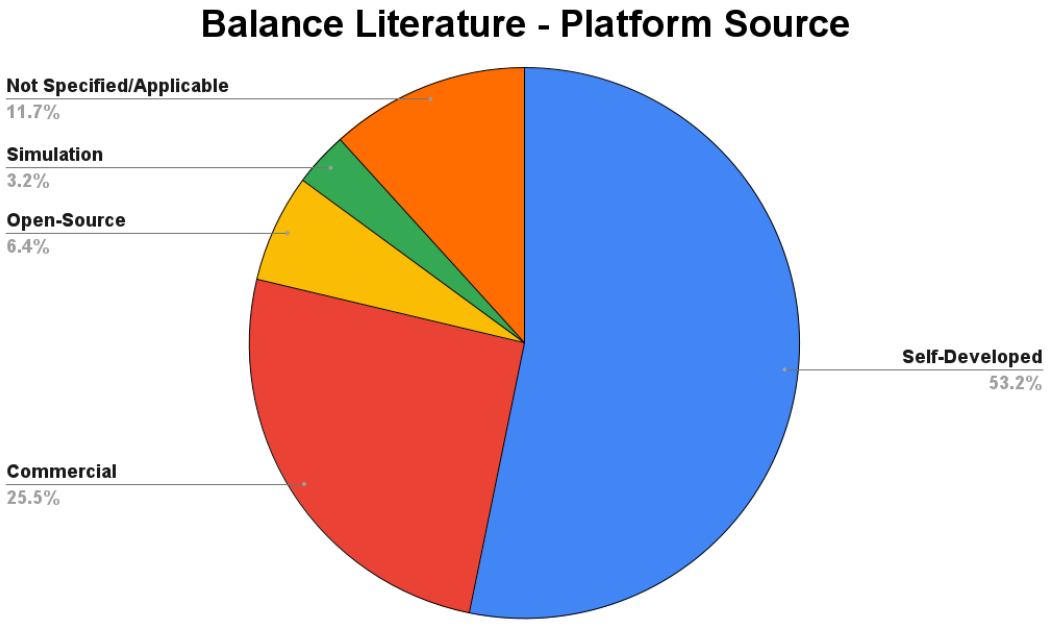

Figure C.2: A graph showing the distribution of publishing platforms used for game balance analysis.

titles such as *Aion* [370, 371], *Pokemon* [390, 389], *Starcraft* [323, 25], and *League of Legends* [245, 222]. Due to availability and access to data and backend systems, there was also an adoption of existing open-source games such as *Neural MMO* [410, 409] and *microRTS* [104]. A small minority simulated game balance without a game or genre specified, attempting to approximate an underlying system [409, 342, 510]. While the platform and genre of game is critical and the evaluated publications had a wide range of applications, authors still converged on a relatively even distribution of balancing themes. This suggests that while genre and platform must be considered in the context of a game, our provided taxonomy is flexible enough to cover a wide variety of games, platforms, and simulations.

Bibliography

[1] Smash Ultimate's 2nd Official Tier List - Luminosity — luminosity.gg. `https://luminosity.gg/news/smash-ultimates-2nd-official-tier-list`, 2024. [Accessed 13-09-2024].

[2] Theodore Agriogianis. The roles, mechanics, and evolution of boss battles in video games. 2018.

[3] Zarif Bin Akhtar. Unveiling the evolution of generative ai (gai): a comprehensive and investigative analysis toward llm models (2021–2024) and beyond. *Journal of Electrical Systems and Information Technology*, 11(1):22, 2024.

[4] Nader Akoury, Qian Yang, and Mohit Iyyer. A framework for exploring player perceptions of llm-generated dialogue in commercial video games. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2295–2311, 2023.

[5] Seyed Hossein Alavi, Weijia Xu, Nebojsa Jojic, Daniel Kennett, Raymond T Ng, Sudha Rao, Haiyan Zhang, Bill Dolan, and Vered Shwartz. Game plot design

with an llm-powered assistant: An empirical study with game designers. *arXiv preprint arXiv:2411.02714*, 2024.

[6] Helmut Alt and Michael Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.

[7] David Altimira, Florian'Floyd' Mueller, Gun Lee, Jenny Clarke, and Mark Billinghurst. Towards understanding balancing in exertion games. In *Proceedings of the 11th conference on advances in computer entertainment technology*, pages 1–8, 2014.

[8] David Altimira, Jenny Clarke, Gun Lee, Mark Billinghurst, Christoph Bartneck, et al. Enhancing player engagement through game balancing in digitally augmented physical games. *International Journal of Human-Computer Studies*, 103: 35–47, 2017.

[9] Frank Ambriz. Deal with it: The challenges of game balancing. `https://frankt1000.wordpress.com/2013/05/31/competitive-gaming-the-challenges-of-game-balancing/`, 2013. [Accessed 13-09-2024].

[10] Cristina Ampatzidou and Katharina Gugerell. Participatory game prototyping–balancing domain content and playability in a serious game design for the energy transition. *CoDesign*, 2019.

[11] Erik Andersen, Sumit Gulwani, and Zoran Popovic. A trace-based framework for analyzing and synthesizing educational progressions. In *Proceedings of the*

SIGCHI Conference on Human Factors in Computing Systems, pages 773–782, 2013.

[12] Barrett R Anderson, Jash Hemant Shah, and Max Kreminski. Homogenization effects of large language models on human creative ideation. In *Proceedings of the 16th Conference on Creativity & Cognition*, pages 413–425, 2024.

[13] Gustavo Andrade, Geber Ramalho, Hugo Santana, and Vincent Corruble. Automatic computer game balancing: a reinforcement learning approach. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 1111–1112, 2005.

[14] Gustavo Andrade, Geber Ramalho, Hugo Santana, and Vincent Corruble. Challenge-sensitive action selection: an application to game balancing. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, pages 194–200. IEEE, 2005.

[15] Gustavo Andrade, Geber Ramalho, Hugo Santana, and Vincent Corruble. Extending reinforcement learning to provide dynamic game balancing. In *Proceedings of the Workshop on Reasoning, Representation, and Learning in Computer Games, 19th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 7–12, 2005.

[16] Gustavo Andrade, Geber Ramalho, Alex Gomes, and Vincent Corruble. Dynamic game balancing: An evaluation of user satisfaction. In *Proceedings of the AAAI*

Conference on Artificial Intelligence and Interactive Digital Entertainment, volume 2, pages 3–8, 2006.

[17] Cameron Angliss, Jiaxun Cui, Jiaheng Hu, Arrasy Rahman, and Peter Stone. A benchmark for generalizing across diverse team strategies in competitive pokémon, 2025. URL `https://arxiv.org/abs/2506.10326`.

[18] Asad Anjum, Yuting Li, Noelle Law, Megan Charity, and Julian Togelius. The ink splotch effect: A case study on chatgpt as a co-creative game designer. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, pages 1–15, 2024.

[19] Alissa N Antle and Alyssa F Wise. Getting down to details: Using theories of cognition and learning to inform tangible user interface design. *Interacting with Computers*, 25(1):1–20, 2013.

[20] Naoto Aoki, Naoki Mori, and Makoto OKada. Analysis of llm-based narrative generation using the agent-based simulation. In *2023 15th International Congress on Advanced Applied Informatics Winter (IIAI-AAI-Winter)*, pages 284–289. IEEE, 2023.

[21] ArenaNet. Guild wars 2. [DIGITAL], 2012.

[22] Aristotle. *Poetics*. Project Gutenberg, 2008. URL `https://www.gutenberg.org/files/1974/1974-h/1974-h.htm`.

[23] Sylvester Arnab, Theodore Lim, Maira B Carvalho, Francesco Bellotti, Sara De Freitas, Sandy Louchart, Neil Suttie, Riccardo Berta, and Alessandro De Glo-

ria. Mapping learning and game mechanics for serious games analysis. *British Journal of Educational Technology*, 46(2):391–411, 2015.

[24] Daniel Atorf, Ehm Kannegieser, and Wolfgang Roller. Balancing realism and engagement for a serious game in the domain of remote sensing. In *Games and Learning Alliance: 7th International Conference, GALA 2018, Palermo, Italy, December 5–7, 2018, Proceedings 7*, pages 146–156. Springer, 2019.

[25] Allan M Avila, Maria Fonoberova, João P Hespanha, Igor Mezić, Daniel Clymer, Jonathan Goldstein, Marco A Pravia, and Daniel Javorsek. Game balancing using koopman-based learning. In *2021 american control conference (acc)*, pages 710–717. IEEE, 2021.

[26] Alexander Baldwin, Daniel Johnson, Peta Wyeth, and Penny Sweetser. A framework of dynamic difficulty adjustment in competitive multiplayer video games. In *2013 IEEE international games innovation conference (IGIC)*, pages 16–19. IEEE, 2013.

[27] Bandai Namco Studios. Super smash bros ultimate. [DIGITAL], 2018.

[28] Sora Ltd. Bandai Namco Studios. Super smash bros. ultimate. [Nintendo Switch], 1987.

[29] Scott Bateman, Regan L Mandryk, Tadeusz Stach, and Carl Gutwin. Target assistance for subtly balancing competitive play. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2355–2364, 2011.

[30] Nicola Baumann, Christoph Lürig, and Stefan Engeser. Flow and enjoyment

beyond skill-demand balance: The role of game pacing curves and personality. *Motivation and Emotion*, 40(4):507–519, 2016.

[31] Philipp Beau and Sander Bakkes. Automated game balancing of asymmetric video games. In *2016 IEEE conference on computational intelligence and games (CIG)*, pages 1–8. IEEE, 2016.

[32] Alexander Becker and Daniel Görlich. Game balancing—a semantical analysis. In *Workshops at the 2nd International Conference on Applied Informatics*, 2019.

[33] Alexander Becker and Daniel Görlich. What is game balancing?-an examination of concepts. *ParadigmPlus*, 1(1):22–41, 2020.

[34] Samuel C Bellini-Leite. Dual process theory: Embodied and predictive; symbolic and classical. *Frontiers in Psychology*, 13:805386, 2022.

[35] Daniel Benmergui. Storyteller, 2023. URL `https://annapurnainteractive.com/en/games/storyteller`.

[36] Marlene Beyer, Aleksandr Agureikin, Alexander Anokhin, Christoph Laenger, Felix Nolte, Jonas Winterberg, Marcel Renka, Martin Rieger, Nicolas Pflanzl, Mike Preuss, et al. An integrated process for game balancing. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2016.

[37] Andrzej Białecki, Natalia Jakubowska, Paweł Dobrowolski, Piotr Białecki, Leszek Krupiński, Andrzej Szczap, Robert Białecki, and Jan Gajewski. Sc2egset: Starcraft ii esport replay and game-state dataset. *Scientific Data*, 10(1):600, 2023.

[38] Peter R Blake and Katherine McAuliffe. "i had so much it didn't seem fair": Eight-year-olds reject two forms of inequity. *Cognition*, 120(2):215–224, 2011.

[39] Blizzard Entertainment. Diablo iii. Video game [PC/Mac], 2012.

[40] Blizzard Entertainment. Starcraft 2. [DIGITAL], 2010.

[41] Blizzard Entertainment. Overwatch 2. [DIGITAL], 2023.

[42] Ian Bogost. *Persuasive games: The expressive power of videogames*. MIT Press, 2010.

[43] Ian Bogost. The blue shell and its discontents, May 2014. URL `https://www.gamedeveloper.com/design/the-blue-shell-and-its-discontents`.

[44] Iavor Bojinov, Guillaume Saint-Jacques, and Martin Tingley. Avoid the pitfalls of a/b testing make sure your experiments recognize customers' varying needs. *Harvard Business Review*, 98(2):48–53, 2020.

[45] Michael Booth. The ai systems of left 4 dead. In *Artificial Intelligence and Interactive Digital Entertainment Conference at Stanford, 2009*, 2009.

[46] David Braben and Ian Bell. Elite, 1984. URL `http://www.elitehomepage.org/`. Video game.

[47] Alexander Brazie. Enemy design in games: A beginner's guide, Oct 2024. URL `https://gamedesignskills.com/game-design/enemy-design/`.

[48] Alexander Brazzie. Video Game Balance: A Definitive Guide — gamedesign-

skills.com. https://gamedesignskills.com/game-design/game-balance/#why-is-it-important-to-balance-a-game. [Accessed 13-09-2024].

[49] Simon Brislin. Balancing Narrative And Gameplay — gamedeveloper.com. https://www.gamedeveloper.com/design/balancing-narrative-and-gameplay, 2013. [Accessed 12-06-2025].

[50] Sarah F Brosnan. Nonhuman species' reactions to inequity and their implications for fairness. *Social justice research*, 19(2):153–185, 2006.

[51] Sarah F Brosnan and Frans BM De Waal. Monkeys reject unequal pay. *Nature*, 425(6955):297–299, 2003.

[52] Mark Brown and Sky LaRell Anderson. Designing for disability: Evaluating the state of accessibility design in video games. *Games and Culture*, 16(6):702–718, 2021.

[53] Nathan Brown. Elden Ring is massively imbalanced, but balance is overrated anwyay — eurogamer.net. https://www.eurogamer.net/elden-ring-is-massively-imbalanced-but-balance-is-overrated-anwyay, 2022. [Accessed 13-09-2024].

[54] Steven Brown and Carmen Tu. The shapes of stories: A "resonator" model of plot structure. *Frontiers of Narrative Studies*, 6(2):259–288, 2020.

[55] Cameron Browne and Frederic Maire. Evolutionary game design. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(1):1–16, 2010.

[56] Bungie. Halo: Combat evolved, 2001. URL https://en.wikipedia.org/wiki/Halo:_Combat_Evolved.

[57] Bungie. Destiny 2. [DIGITAL], 2017.

[58] Steph Buongiorno, Lawrence Jake Klinkert, Tanishq Chawla, Zixin Zhuang, and Corey Clark. Pangea: Procedural artificial narrative using generative ai for turn-based video games. *arXiv preprint arXiv:2404.19721*, 2024.

[59] Haley Burch and Newton Lee. *Pokémon and World Championships*, pages 1431–1435. Springer International Publishing, Cham, 2024. ISBN 978-3-031-23161-2. doi: 10.1007/978-3-031-23161-2_475. URL https://doi.org/10.1007/978-3-031-23161-2_475.

[60] Keith Burgun. Understanding Balance in Video Games — gamedeveloper.com. https://www.gamedeveloper.com/design/understanding-balance-in-video-games, 2011. [Accessed 13-09-2024].

[61] Andrew Burnes. Nvidia digital human technologies bring ai game characters to life, March 2024. URL https://www.nvidia.com/en-us/geforce/news/nvidia-ace-gdc-gtc-2024-ai-character-game-and-app-demo-videos/.

[62] Luis Javier Cabeza-Ramírez, Guzmán Antonio Muñoz-Fernández, and Luna Santos-Roldán. Video game streaming in young people and teenagers: Uptake, user groups, dangers, and opportunities. In *Healthcare*, volume 9, page 192. MDPI, 2021.

[63] Alex Calderwood, John Joon Young Chung, Yuqian Sun, Melissa Roemmele,

and Max Kreminski. Phraselette: A poet's procedural palette. *arXiv preprint arXiv:2503.06335*, 2025.

[64] Janelynn Camingue, Edward F Melcer, and Elin Carstensdottir. A (visual) novel route to learning: a taxonomy of teaching strategies in visual novels. In *Proceedings of the 15th International Conference on the Foundations of Digital Games*, pages 1–13, 2020.

[65] Janelynn Camingue, Elin Carstensdottir, and Edward F Melcer. What is a visual novel? *Proceedings of the ACM on Human-Computer Interaction*, 5(CHI PLAY): 1–18, 2021.

[66] Capcom Co. Ltd. Street fighter, 1987.

[67] Capcom Production Studio 4. Resident evil 4. [Gamecube], 2005.

[68] Loïc Caroux and Katherine Isbister. Influence of head-up displays' characteristics on user experience in video games. *International Journal of Human-Computer Studies*, 87:65–79, 2016.

[69] Justin T Carter. *A conceptual framework of game feel: An evolutionary approach.* PhD thesis, Queensland University of Technology, 2022.

[70] Polona Caserman, Katrin Hoffmann, Philipp M¨uller, Marcel Schaub, Katharina Straßburg, Josef Wiemeyer, Regina Bruder, Stefan G¨obel, et al. Quality criteria for serious games: serious part, game part, and balance. *JMIR serious games*, 8 (3):e19037, 2020.

[71] Katharine Castle. Tunic's instruction manual and the zelda art that inspired it. *Rock Paper Shotgun*, March 2022. URL `https://www.rockpapershotgun.com/tunics-instruction-manual-and-the-zelda-art-that-inspired-it`.

[72] Fadi Castronovo, Robert M Leicht, and John I Messner. When is a construction educational serious game too serious? striking a balance between engagement and learning. In *Computing in Civil Engineering 2017*, pages 26–34. 2017.

[73] Jared E Cechanowicz, Carl Gutwin, Scott Bateman, Regan Mandryk, and Ian Stavness. Improving player balancing in racing games. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 47–56, 2014.

[74] Darryl Charles, Dominic Holmes, Therese Charles, and Suzanne McDonough. Virtual reality design for stroke rehabilitation. *Biomedical Visualisation: Volume 6*, pages 53–87, 2020.

[75] K. Chauhan and A. Chakrabarti. Managing uncertainty in the design and development process by appropriate methods selection. In *Proceedings of the 19th International Conference on Engineering Design (ICED13)*. The Design Society, 2013.

[76] Jenova Chen. Flow in games (and everything else). *Communications of the ACM*, 50(4):31–43, 2007.

[77] Jenova Chen. Designing journey. Talk at Game Developers Conference (GDC)

2013; GDC Vault video, 2013. URL https://gdcvault.com/play/1017700/Designing. Design track; thatgamecompany.

[78] Mingliu Chen, Adam N Elmachtoub, and Xiao Lei. Matchmaking strategies for maximizing player engagement in video games. *Available at SSRN 3928966*, 2021.

[79] Jinghui Cheng, Cynthia Putnam, and Jin Guo. "" always a tall order"" values and practices of professional game designers of serious games for health. In *Proceedings of the 2016 annual symposium on computer-human interaction in play*, pages 217–228, 2016.

[80] Andrew A Chien, Liuzixuan Lin, Hai Nguyen, Varsha Rao, Tristan Sharma, and Rajini Wijayawardana. Reducing the carbon impact of generative ai inference (today and in 2035). In *Proceedings of the 2nd workshop on sustainable computer systems*, pages 1–7, 2023.

[81] Hyerim Cho, Andy Donovan, and Jin Ha Lee. Art in an algorithm: A taxonomy for describing video game visual styles. *Journal of the Association for Information Science and Technology*, 69(5):633–646, 2018.

[82] Sahili Choudhury, Akiko Fujita, and Bernie Lo. CEO of video game maker Nexon flags the 'single most important idea' in his industry — cnbc.com. https://www.cnbc.com/2018/03/19/video-games-nexon-ceo-says-longevity-is-critical-for-video-games.html, 2018. [Accessed 13-09-2024].

[83] Thomas W Christ. Scientific-based research and randomized controlled trials, the

"gold" standard? alternative paradigms and mixed methodologies. *Qualitative Inquiry*, 20(1):72–80, 2014.

[84] Victoria Clarke and Virginia Braun. Thematic analysis. *The journal of positive psychology*, 12(3):297–298, 2017.

[85] Julien Codsi and Adrian Vetta. A case study in learning in metagames: Super smash bros. melee. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 17(1):2–9, Oct. 2021. URL `https://ojs.aaai.org/index.php/AIIDE/article/view/18884`.

[86] Simon Colton and Cameron Browne. Evolving simple art-based games. In *Workshops on Applications of Evolutionary Computation*, pages 283–292. Springer, 2009.

[87] Kate Compton. So you want to build a generator... `http://galaxykate0.tumblr.com/post/139774965871/so-you-want-to-build-a-generator`, Feb 2016. Tumblr blog post; original source of the "1000 bowls of oatmeal" concept.

[88] Michael Cook. Game design is generative design. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 21, pages 22–31, 2025.

[89] Michael Cook, Simon Colton, and Jeremy Gow. The angelina videogame design system—part i. *IEEE Transactions on Computational Intelligence and AI in Games*, 9(2):192–203, 2016.

[90] Michael Cook, Simon Colton, and Jeremy Gow. The angelina videogame design

system—part ii. *IEEE Transactions on Computational Intelligence and AI in Games*, 9(3):254–266, 2016.

[91] Michael Cook, M Charity, Maren Awiszus, Filippo Carnovalini, and Alexander Dockhorn. We call this controller skip: Ai for speedrunning. In *Proceedings of the Workshop on Experimental AI in Games*, 2025.

[92] Siobhán Corrigan, GDR Zon, A Maij, Nick McDonald, and L Mårtensson. An approach to collaborative learning and the serious game development. *Cognition, Technology & Work*, 17:269–278, 2015.

[93] Greg Costikyan. *Uncertainty in games*. Mit Press, 2013.

[94] Emily Crawford. Glitch horror: Ben drowned and the fallibility of technology in game fan fiction. *Transactions of the Digital Games Research Association*, 4(2), 2018.

[95] Sarah C Cregan, Adam J Toth, and Mark J Campbell. Playing for keeps or just playing with emotion? studying tilt and emotion regulation in video games. *Frontiers in psychology*, 15:1385242, 2024.

[96] Timothy Crick. The game body: Toward a phenomenology of contemporary video gaming. *Games and Culture*, 6(3):259–269, 2011.

[97] Marjorie Ann M Cuerdo and Edward F Melcer. " i'll be back": A taxonomy of death and rebirth in platformer video games. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[98] Joe Cutting, Sebastian Deterding, Simon Demediuk, and Nick Sephton. Difficulty-skill balance does not affect engagement and enjoyment: a pre-registered study using artificial intelligence-controlled difficulty. *Royal Society Open Science*, 10 (2):220274, 2023.

[99] Dan Fornance. Rivals of aether, 2017.

[100] Daybreak Game Company. Everquest ii. Video Game, 2004. URL `https://www.everquest2.com/`. Originally developed by Sony Online Entertainment.

[101] Fernando de Mesentier Silva, Rodrigo Canaan, Scott Lee, Matthew C Fontaine, Julian Togelius, and Amy K Hoover. Evolving the hearthstone meta. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2019.

[102] Mathijs De Vaan, David Stark, and Balazs Vedres. Game changer: The topology of creativity. *American Journal of Sociology*, 120(4):1144–1194, 2015.

[103] Michael S Debus. Metagames: on the ontology of games outside of games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–9, 2017.

[104] Daniel A DeLaurentis, Jitesh H Panchal, Ali K Raz, Prajwal Balasubramani, Apoorv Maheshwari, Adam Dachowicz, and Kshitij Mall. Toward automated game balance: A systematic engineering design approach. In *2021 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2021.

[105] Iris Delikoura, Yi R Fung, and Pan Hui. From superficial outputs to super-

ficial learning: Risks of large language models in education. *arXiv preprint arXiv:2509.21972*, 2025.

[106] Kody R Dillman, Terrance Tin Hoi Mok, Anthony Tang, Lora Oehlberg, and Alex Mitchell. A visual interaction cue framework from video game environments for augmented reality. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pages 1–12, 2018.

[107] N. Djafarova, A. Dimitriadou, L. Zefi, and O. Turetken. The art of serious game design: A framework and methodology. *AIS Transactions on Human-Computer Interaction*, 15(3):322–349, 2023.

[108] Paul Dourish. *Where the action is: the foundations of embodied interaction*. MIT press, 2001.

[109] Anders Drachen and Alessandro Canossa. Analyzing spatial user behavior in computer games using geographic information systems. In *Proceedings of the 13th international MindTrek conference: Everyday life in the ubiquitous era*, pages 182–189, 2009.

[110] Anders Drachen and Alessandro Canossa. Towards gameplay analysis via gameplay metrics. In *Proceedings of the 13th international MindTrek conference: Everyday life in the ubiquitous era*, pages 202–209, 2009.

[111] Anders Drachen, Magy Seif El-Nasr, and Alessandro Canossa. Game analytics–the basics. In *Game analytics: Maximizing the value of player data*, pages 13–40. Springer, 2013.

[112] Hubert L Dreyfus. A phenomenology of skill acquisition as the basis for a merleau-pontian nonrepresentational cognitive science. 2002.

[113] Dagmara Dziedzic and Wojciech Włodarczyk. Approaches to measuring the difficulty of games in dynamic difficulty adjustment systems. *International Journal of Human–Computer Interaction*, 34(8):707–715, 2018.

[114] Nintendo EAD. Mario kart 8. [DIGITAL], 2014.

[115] Marc Ebner, John Levine, Simon M Lucas, Tom Schaul, Tommy Thompson, and Julian Togelius. Towards a video game description language. *Artificial and Computational Intelligence in Games*, page 85, 2013.

[116] John Edwards. Sand Rendering in Journey. `https://gdcvault.com/play/1017742/Sand-Rendering-in`, 2013. GDC Vault, Visual Arts Track.

[117] Ben Egliston. Multimodality and the competitive metagame: Exploring issues of balance in multimodal game environments. In *Proceedings of the 2014 Conference on Interactive Entertainment*, pages 1–4, 2014.

[118] Nico Elbert, Christoph M. Flath, Victor Klockmann, Fabian Kosse, Alicia von Schenk, and Nikolai Stein. Matchmaking beyond elo: Why teams are more than the sum of individual skill. In *2025 IEEE Conference on Games (CoG)*, pages 1–8, 2025. doi: 10.1109/CoG64752.2025.11114265.

[119] Embark Studios. Arc raiders, 2025. URL `https://en.wikipedia.org/wiki/ARC_Raiders`.

[120] Barrett Ens, Juan David Hincapié-Ramos, and Pourang Irani. Ethereal planes: a design framework for 2d information space in 3d mixed reality environments. In *Proceedings of the 2nd ACM symposium on Spatial user interaction*, pages 2–12, 2014.

[121] Blizzard Entertainment. Hearthstone. [Digital], 2014.

[122] Steven W Evans, Theodore P Beauchaine, Andrea Chronis-Tuscano, Stephen P Becker, Anil Chacko, Richard Gallagher, Cynthia M Hartung, Michael J Kofler, Brandon K Schultz, Leanne Tamm, et al. The efficacy of cognitive videogame training for adhd and what fda clearance means for clinicians. *Evidence-Based Practice in Child and Adolescent Mental Health*, 6(1):116–130, 2021.

[123] Shih-Wei Fang and Sai-Keung Wong. Game team balancing by using particle swarm optimization. *Knowledge-Based Systems*, 34:91–96, 2012.

[124] Dan Felder. Design 101: Balancing Games — gamedeveloper.com. `https://www.gamedeveloper.com/design/design-101-balancing-games`, 2015. [Accessed 13-09-2024].

[125] Clara Fernández-Vara. *Introduction to Game Analysis*. Routledge, 2019. Referenced for Puzzle Dependency Graphs.

[126] Felipe Machado Figueira, Lucas Nascimento, Jose da Silva Junior, Troy Kohwalter, Leonardo Murta, and Esteban Clua. Bing: A framework for dynamic game balancing using provenance. In *2018 17th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, pages 57–5709. IEEE, 2018.

[127] Francesca Foffano and David Thue. Changes of user experience in an adaptive game: a study of an ai manager. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, pages 1–8, 2019.

[128] Eelke Folmer. Component based game development–a solution to escalating costs and expanding deadlines? In *International symposium on component-based software engineering*, pages 66–73. Springer, 2007.

[129] Dom Ford. Giantness and excess in dark souls. In *Proceedings of the 15th international conference on the foundations of digital games*, pages 1–4, 2020.

[130] Anders Frank. Balancing three different foci in the design of serious games: Engagement, training objective and context. In *Proceedings of DiGRA 2007 Conference: Situated Play*, 2007.

[131] Christopher Franzwa, Ying Tang, and Aaron Johnson. Serious game design: Motivating students through a balance of fun and learning. In *2013 5th International conference on games and virtual worlds for serious applications (VS-GAMES)*, pages 1–7. IEEE, 2013.

[132] Alexander Freed. Branching conversation systems and the working writer, part 2: Design considerations, Sep 2014. URL `https://www.gamedeveloper.com/design/branching-conversation-systems-and-the-working-writer-part-2-design-considerations`.

[133] Peter W Frey and Peter Adesman. Recall memory for visually presented chess positions. *Memory & Cognition*, 4(5):541–547, 1976.

[134] Jane Friedhoff. Untangling twine: A platform study. In *Proceedings of DiGRA 2013 Conference*, Tampere, 2014. DiGRA.

[135] FromSoftware. Dark souls. [Playstation 3, Xbox 360, Windows, Playstation 4, Xbox One, Nintendo Switch], 2011.

[136] FromSoftware. Elden ring. [DIGITAL], 2022.

[137] Tracy Fullerton. *Game design workshop: a playcentric approach to creating innovative games*. AK Peters/CrC Press, 2024.

[138] Tracy Fullerton, Chris Swain, and Steven Hoffman. *Game design workshop: Designing, prototyping, & playtesting games*. CRC Press, 2004.

[139] Shaun Gallagher. What is phenomenology? In *Phenomenology*, pages 1–10. Springer, 2022.

[140] Vittorio Gallese. Embodied simulation: from mirror neuron systems to interpersonal relations. In *Empathy and Fairness: Novartis Foundation Symposium 278*, pages 3–19. Wiley Online Library, 2006.

[141] Roberto Gallotta, Graham Todd, Marvin Zammit, Sam Earle, Antonios Liapis, Julian Togelius, and Georgios N Yannakakis. Large language models and games: A survey and roadmap. *arXiv preprint arXiv:2402.18659*, 2024.

[142] Game Freak. Pokemon scarlet and violet. [DIGITAL], 2022.

[143] Insomniac Games. Spyro the dragon, 1998. URL `https://en.wikipedia.org/wiki/Spyro_the_Dragon`.

[144] Jacob Garbe, Max Kreminski, Ben Samuel, Noah Wardrip-Fruin, and Michael Mateas. Storyassembler: an engine for generating dynamic choice-driven narratives. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, FDG '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450372176. doi: 10.1145/3337722.3337732. URL https://doi.org/10.1145/3337722.3337732.

[145] M Gardner. Mathematical games: The fantastic combinations of john conway's new solitaire game'life'scientific american, 1970.

[146] Jesse James Garrett. *The Elements of User Experience: User-Centered Design for the Web and Beyond*. Pearson Education, 2010.

[147] Aron Garst. Super smash bros. fans have been tripping over themselves for 15 years. *GameSpot*, March 2023. URL https://www.gamespot.com/articles/super-smash-bros-fans-have-been-tripping-over-themselves-for-15-years/1100-6512174/. Retrospective analysis of the random tripping mechanic and its negative impact on the competitive community.

[148] James Paul Gee. Video games and embodiment. *Games and culture*, 3(3-4): 253–263, 2008.

[149] A Shaji George, AS Hovan George, and AS Gabrio Martin. The environmental impact of ai: a case study of water consumption by chat gpt. *Partners Universal International Innovation Journal*, 1(2):97–104, 2023.

[150] Kathrin Maria Gerling, Matthew Miller, Regan L Mandryk, Max Valentin Birk,

and Jan David Smeddinck. Effects of balancing for physical abilities on player performance, experience and self-esteem in exergames. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2201–2210, 2014.

[151] Emily Gertenbach. The most in-demand skills and jobs for 2024, June 2024. URL https://www.upwork.com/resources/in-demand-jobs-and-skills.

[152] Ghost Ship Games. Deep rock galactic. [DIGITAL], 2020.

[153] Anthony Giovannetti. 'slay the spire': Metrics driven design and balance. GDC Vault, Mar 2019. URL https://www.gdcvault.com/play/1025731/-Slay-the-Spire-Metrics. Game Developers Conference (GDC) 2019.

[154] Matthew E Gladden. *Phenomenology of the Gameworld: A Philosophical Toolbox for Video Game Developers*. Defragmenter Media, 2019.

[155] Madelyn Glymour. Libguides: Qualitative research methods: Phenomenology. URL https://guides.library.duq.edu/c.php?g=836228&p=5972144#:~:text=A%20phenomenological%20study%20explores%20what,before%20embarking%20on%20your%20research.

[156] Vinod Goel and Peter Pirolli. The structure of design problem spaces. *Cognitive Science*, 16(3):395–429, 1992.

[157] David Gonçalves, Daniel Barros, Pedro Pais, João Guerreiro, Tiago Guerreiro, and André Rodrigues. The trick is to stay behind?: Defining and exploring the design space of player balancing mechanics. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2024.

[158] Justin Gordon. Masahiro Sakurai explains how the 'obvious solution' for balancing character traits will usually lead to a game's downfall — eventhubs.com. https://www.eventhubs.com/news/2024/aug/31/masahiro-sakurai-balance-solution-traits/, 2024. [Accessed 13-09-2024].

[159] Jasmine Gould-Wilson. 'We've learned to love and trust the process': How Hades 2 built on Supergiant's Early Access legacy to deliver the best roguelike of 2025. *GamesRadar+*, dec 2025. URL https://www.gamesradar.com/games/hades/weve-learned-to-love-and-trust-the-process-how-hades-2-built-on-supergiants-early-access-legacy-to-deliver-the-best-roguelike-of-2025/.

[160] Rez Graham. The human cost of generative ai. Game AI Summit. Game Developers Conference, March 2025.

[161] Nora Graves, Vitus Larrieu, Yingyue Trace Zhang, Joanne Peng, Varun Nagaraj Rao, Yuhan Liu, and Andrés Monroy-Hernández. Gptfootprint: Increasing consumer awareness of the environmental impacts of llms. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2025.

[162] Michael Cerny Green, Ahmed Khalifa, Athoug Alsoughayer, Divyesh Surana, Antonios Liapis, and Julian Togelius. Two-step constructive approaches for dungeon generation. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, pages 1–7, San Luis Obispo California USA, August

2019. ACM. ISBN 978-1-4503-7217-6. doi: 10.1145/3337722.3341847. URL https://dl.acm.org/doi/10.1145/3337722.3341847.

[163] Joshua Greene and Jonathan Haidt. How (and where) does moral judgment work? *Trends in cognitive sciences*, 6(12):517–523, 2002.

[164] Joshua D Greene, Leigh E Nystrom, Andrew D Engell, John M Darley, and Jonathan D Cohen. The neural bases of cognitive conflict and control in moral judgment. *Neuron*, 44(2):389–400, 2004.

[165] Zhiwei Guan, Shirley Lee, Elisabeth Cuddihy, and Judith Ramey. The validity of the stimulated retrospective think-aloud method as measured by eye tracking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*, pages 1253–1262, New York, NY, USA, 2006. ACM. doi: 10.1145/1124772.1124961.

[166] Eyjolfur Gudmundsson and Einar Sigurdur Hreidarson. The art of war: Effective ways to address the rmt issue. In *GDC Vault*, 2010. URL https://gdcvault.com/play/1012260/The-Art-of-War-Effective.

[167] EAA Gunn, BGW Craenen, and E Hart. A taxonomy of video games and ai. In *AI and Games Symposium*, pages 4–14. Citeseer, 2009.

[168] Pablo Gutiérrez-Sánchez, Diego Pérez-Liébana, and Raluca D Gaina. Explaining and clustering playtraces using temporal logics. In *Proceedings of the 20th International Conference on the Foundations of Digital Games*, FDG '25, New York,

NY, USA, 2025. Association for Computing Machinery. ISBN 9798400718564. doi: 10.1145/3723498.3723719. URL https://doi.org/10.1145/3723498.3723719.

[169] Matthew Guzdial, Nicholas Liao, and Mark Riedl. Co-Creative Level Design via Machine Learning. 2018. doi: 10.48550/ARXIV.1809.09420. URL https://arxiv.org/abs/1809.09420. Publisher: arXiv Version Number: 1.

[170] Matthew Guzdial, Devi Acharya, Max Kreminski, Michael Cook, Mirjam Eladhari, Antonios Liapis, and Anne Sullivan. Tabletop Roleplaying Games as Procedural Content Generators. In *International Conference on the Foundations of Digital Games*, FDG '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 978-1-4503-8807-8. doi: 10.1145/3402942.3409605. URL https://doi.org/10.1145/3402942.3409605. event-place: Bugibba, Malta.

[171] Gary Gygax, Richard Garfield, Greg Costikyan, Marc Miller, Matt Forbeck, Greg Stafford, and Rick Loomis. *Horsemen of the Apocalypse: Essays on Roleplaying.* Jolly Roger Games, 2000. ISBN 1558782400.

[172] Chad Habel and Ben Kooyman. Agency mechanics: gameplay design in survival horror video games. *Digital Creativity*, 25(1):1–14, 2014.

[173] HAL Laboratory/Nintendo. Super smash bros. melee, 2001.

[174] Joshua V Hall, Peta A Wyeth, and Daniel Johnson. Instructional objectives to core-gameplay: a serious game design technique. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 121–130, 2014.

[175] Patricia L Hardré. When, how, and why do we trust technology too much? In *Emotions, technology, and behaviors*, pages 85–106. Elsevier, 2016.

[176] John Harris, Mark Hancock, and Stacey D Scott. Leveraging asymmetries in multiplayer games: Investigating design elements of interdependent play. In *Proceedings of the 2016 Annual Symposium on computer-human interaction in play*, pages 350–361, 2016.

[177] Casper Harteveld. *Triadic Game Design: Balancing Reality, Meaning and Play.* Springer Science & Business Media, 2011.

[178] Casper Harteveld, Rui Guimarães, Igor Mayer, and Rafael Bidarra. Balancing pedagogy, game and reality components within a unique serious game for training levee inspection. In *Technologies for E-Learning and Digital Entertainment: Second International Conference, Edutainment 2007, Hong Kong, China, June 11-13, 2007. Proceedings 2*, pages 128–139. Springer, 2007.

[179] Emma Harvey, Allison Koenecke, and Rene F Kizilcec. " don't forget the teachers": Towards an educator-centered understanding of harms from large language models in education. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–19, 2025.

[180] Erin J Hastings, Ratan K Guha, and Kenneth O Stanley. Evolving content in the galactic arms race video game. In *2009 IEEE Symposium on Computational Intelligence and Games*, pages 241–248. IEEE, 2009.

[181] Erin Jonathan Hastings, Ratan K Guha, and Kenneth O Stanley. Automatic

content generation in the galactic arms race video game. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(4):245–263, 2009.

[182] Guy Hawkins, Keith Nesbitt, and Scott Brown. Dynamic difficulty balancing for cautious players and risk takers. *International Journal of Computer Games Technology*, 2012(1):625476, 2012.

[183] Michael Hemmingsen. Code is law: Subversion and collective knowledge in the ethos of video game speedrunning. *Sport, Ethics and Philosophy*, 15(3):435–460, 2021.

[184] Maurice Hendrix, Tyrone Bellamy-Wood, Sam McKay, Victoria Bloom, and Ian Dunwell. Implementing adaptive game difficulty balancing in serious games. *IEEE Transactions on Games*, 11(4):320–327, 2018.

[185] Anna Hernandelius. Leveling up: How AI's transformative role in level automation production adds business value in 'Candy Crush Saga'. Game Developers Conference (GDC) Vault, 2024. URL `https://gdcvault.com/play/1034631/Leveling-Up-How-AI-s`. Presented at GDC 2024.

[186] Daniel Hernandez, Charles Takashi Toyin Gbadamosi, James Goodman, and James Alfred Walker. Metagame autobalancing for competitive multiplayer games. In *2020 IEEE Conference on Games (CoG)*, pages 275–282. IEEE, 2020.

[187] Kieran Hicks, Patrick Dickinson, Jussi Holopainen, and Kathrin Gerling. Good game feel: an empirically grounded framework for juicy design. In *Proceedings of DiGRA 2018 Conference: The Game is the Message*, 2018.

[188] Larissa Hjorth and Dean Chan. *Gaming cultures and place in Asia-Pacific*, volume 5. Routledge, 2009.

[189] Vincent Hom and Joe Marks. Automatic design of balanced board games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 3, pages 25–30, 2007.

[190] Geoffrey Hookham, Bridgette Bewick, Frances Kay-Lambkin, Andrew Healey, and Kristy Monaghan. A concurrent think aloud study of engagement and usability in a serious game. In *Proceedings of the Second Joint International Conference on Serious Games (JCSG)*, pages 3–15, Brisbane, Australia, 2016. Springer. doi: 10.1007/978-3-319-45841-0_1.

[191] Jettie Hoonhout. Let the game tester do the talking: Think aloud and interviewing to learn about the game experience. In *Game usability*, pages 115–125. CRC Press, 2022.

[192] Danial Hooshyar, Moslem Yousefi, and Heuiseok Lim. Data-driven approaches to game player modeling: a systematic literature review. *ACM Computing Surveys (CSUR)*, 50(6):1–19, 2018.

[193] Ian Horswill. Step: a highly expressive text generation language. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, pages 240–249, 2022.

[194] Peyman Hosseini, Ignacio Castro, Iacopo Ghinassi, and Matthew Purver. Efficient solutions for an intriguing failure of llms: Long context window does not mean

llms can analyze long sequences flawlessly, 2024. URL https://arxiv.org/abs/2408.01866.

[195] Qiangru Huang, Junqing Lin, Rui Han, Cheng Peng, and Aji Huang. Using virtual reality exposure therapy in pain management: a systematic review and meta-analysis of randomized controlled trials. *Value in Health*, 25(2):288–301, 2022.

[196] Hudson Soft. Mario party. [Nintendo 64], 1998.

[197] Kenneth Hullett and Jim Whitehead. Design patterns in fps levels. In *proceedings of the Fifth International Conference on the Foundations of Digital Games*, pages 78–85, 2010.

[198] Robin Hunicke. The case for dynamic difficulty adjustment in games. In *Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 429–433, 2005.

[199] Robin Hunicke, Marc LeBlanc, Robert Zubek, et al. Mda: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI*, volume 4, page 1722. San Jose, CA, 2004.

[200] Daeun Hwang, Samuel Shields, Alex Calderwood, Shi Johnson-Bey, Michael Mateas, Noah Wardrip-Fruin, and Edward F. Melcer. "clicking some of the silly options": Exploring player motivation in static and dynamic educational interactive narratives, 2025. Workshop paper, CHI 2025: Augmented Educators and AI.

[201] Susan Hwang, Adrian L Jessup Schneider, Daniel Clarke, Alexander Macintosh, Lauren Switzer, Darcy Fehlings, and TC Nicholas Graham. How game balancing affects play: Player adaptation in an exergame for children with cerebral palsy. In *Proceedings of the 2017 Conference on Designing Interactive Systems*, pages 699–710, 2017.

[202] Sami Hyrynsalmi, Eriks Klotins, Michael Unterkalmsteiner, Tony Gorschek, Nirnaya Tripathi, Leandro Bento Pompermaier, and Rafael Prikladnicki. What is a minimum viable (video) game? towards a research agenda. In *Conference on e-Business, e-Services and e-Society*, pages 217–231. Springer, 2018.

[203] Ioanna Iacovides, Anna Cox, Richard Kennedy, Paul Cairns, and Charlene Jennett. Removing the hud: the impact of non-diegetic game elements and expertise on player involvement. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play*, pages 13–22, 2015.

[204] id Software. Doom. [DIGITAL], 2016.

[205] Wijnand A IJsselsteijn, Yvonne AW de Kort, and Karolien Poels. The game experience questionnaire: Development of a self-report measure to assess player experiences of digital games. Technical report, Technische Universiteit Eindhoven, Eindhoven, Netherlands, 2013. Widely cited as the standard validation for game experience surveys.

[206] Usama Imtiaz and Hasan Mujtaba. Adaptive ai for competitive gaming: particle-

swarm-optimized neural network for skill, engagement, and strategic evolution. *PeerJ Computer Science*, 11:e3347, 2025.

[207] Informa PLC. Game Developer Search: game balancing. `https://www.gamedeveloper.com/search?q=game+balancing`, 2026. Search for "game balancing" yielded approximately 1,200 articles. [Accessed 07-02-2026].

[208] Informa PLC. GDC Vault Search: "game balance". `https://gdcvault.com/search.php?q="game+balance"`, 2026. Search for "game balance" yielded approximately 450 results. [Accessed 07-02-2026].

[209] inkle. URL `https://www.inklestudios.com/ink/`.

[210] Aaron Isaksen, Daniel Gopstein, and Andrew Nealen. Exploring game space using survival analysis. In *FDG*, 2015.

[211] Katherine Isbister. Enabling social play: A framework for design and evaluation. In *Evaluating user experience in games: Concepts and methods*, pages 11–22. Springer, 2009.

[212] Damian Isla. Handling complexity in the halo 2 ai. In *Proceedings of the Game Developers Conference (GDC)*, volume 12, 2005.

[213] Isto Inc. Get to work, Dec 2024. URL `https://store.steampowered.com/app/2706170/Get_To_Work/`. Video game.

[214] Isto Inc. How speedrunners broke my rage game (get to work). `https://www.youtube.com/watch?v=yIpms96cr1o`, Mar 2025. YouTube video.

[215] Frank Jackson. Epiphenomenal qualia. In *Consciousness and emotion in cognitive science*, pages 197–206. Routledge, 1998.

[216] Alex Jaffe. Cursed problems in game design. GDC Vault, March 2019. URL `https://www.gdcvault.com/play/1025756/Cursed-Problems-in-Game`. Presented at the Game Developers Conference (GDC) 2019.

[217] Alexander Jaffe, Alex Miller, Erik Andersen, Yun-En Liu, Anna Karlin, and Zoran Popovic. Evaluating competitive game balance with restricted play. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 8, pages 26–31, 2012.

[218] Patrick Jagoda. Experimental games: Critique, play, and design in the age of gamification. In *Experimental Games*. University of Chicago Press, 2020.

[219] Monique WM Jaspers, Thiemo Steen, Cor Van Den Bos, and Maud Geenen. The think aloud method: a guide to user interface design. *International journal of medical informatics*, 73(11-12):781–795, 2004.

[220] Jayasankar Jayachandran and Vijay Arni. Traversing the ethical landscape of data scraping for ai. *Available at SSRN 4666354*, 2023.

[221] Hyeon-Chang Jeon, In-Chang Baek, Cheong-mok Bae, Taehwa Park, Wonsang You, Taegwan Ha, Hoyoun Jung, Jinha Noh, Seungwon Oh, and Kyung-Joong Kim. Raidenv: Exploring new challenges in automated content balancing for boss raid games. *IEEE Transactions on Games*, 2023.

[222] Julie Jiang, Danaja Maldeniya, Kristina Lerman, and Emilio Ferrara. The wide,

the deep, and the maverick: Types of players in team-based online games. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW1):1–26, 2021.

[223] Yilin Jiang, Mingzi Zhang, Sheng Jin, Zengyi Yu, Xiangjie Kong, and Binghao Tu. Emnlp: Educator-role moral and normative large language models profiling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 815–843, 2025.

[224] Laura Jiménez-Muñoz, Inmaculada Peñuelas-Calvo, Pilar Calvo-Rivera, Isaac Díaz-Oliván, Manon Moreno, Enrique Baca-García, and Alejandro Porras-Segovia. Video games for the treatment of autism spectrum disorder: A systematic review. *Journal of Autism and Developmental Disorders*, 52(1):169–188, 2022.

[225] Magnus Johansson, Harko Verhagen, Anna Åkerfeldt, and Staffan Selander. How to design for meaningful learning–finding the balance between learning and game components. In *Proceedings of the 8th European conference on games based learning*, pages 216–222, 2014.

[226] Shi Johnson-Bey, Kira Liao, Samuel Shields, Daeun Hwang, Noah Wardrip-Fruin, Michael Mateas, and Edward Melcer. Building visual novels with social simulation and storylets. In *International Conference on Interactive Digital Storytelling*, pages 145–161. Springer, 2024.

[227] Lasse Juel Larsen and Bo Kampmann Walther. The ontology of gameplay: Toward a new theory. *Games and Culture*, 15(6):609–631, 2020.

[228] Junebug. How donkey kong works and how project m overtuned him, July 11 2023. URL https://www.youtube.com/watch?v=yPI5iZ9d4aE.

[229] Wilian Gatti Junior, Emily Marasco, Beaumie Kim, Laleh Behjat, and Marjan Eggermont. How chatgpt can inspire and improve serious board game design. *International Journal of Serious Games*, 10(4):33–54, 2023.

[230] Jesper Juul. *Half-real: Video games between real rules and fictional worlds*. MIT press, 2011.

[231] Jeff Kaplan. OW update/balancing cycle is excruciatingly slow: Part 2 - Overwatch Forums — web.archive.org. https://web.archive.org/web/20170708042628/https://us.battle.net/forums/en/overwatch/topic/20757706588?page=2#post-27, 2017. [Accessed 13-09-2024].

[232] Daniel Karavolos, Antonios Liapis, and Georgios Yannakakis. Learning the patterns of balance in a multi-player shooter game. In *Proceedings of the 12th international conference on the foundations of digital games*, pages 1–10, 2017.

[233] Veli-Matti Karhulahti, Miia Siutila, Jukka Vahlo, and Raine Koskimaa. Phenomenological strands for gaming disorder and esports play: a qualitative registered report. *Collabra: Psychology*, 8(1):38819, 2022.

[234] Greg Kasavin. The chaos update is now available! https://www.supergiantgames.com/blog/hades-the-chaos-update-is-now-available/, Jan 2019. Supergiant Games Blog.

[235] William Kavanagh, Alice Miller, Gethin Norman, and Oana Andrei. Balancing turn-based games with chained strategy generation. *IEEE Transactions on Games*, 13(2):113–122, 2019.

[236] Brendan Keogh. *A play of bodies: How we perceive videogames*. MIT Press, 2018.

[237] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3):358–386, 2005.

[238] Matcheri Keshavan, John Torous, and Walid Yassin. Do generative ai chatbots increase psychosis risk? *World Psychiatry*, 25(1):150, 2026.

[239] David Kessing and Manuel Löwer. Game-balance simulation as a tool for the evaluation of systematically designed gamification strategies. *STARTPLAY 2022*, page 5, 2022.

[240] Artian Kica, Andrew La Manna, Lindsay O'Donnell, Tom Paolillo, and Mark Claypool. Nerfs, buffs and bugs-analysis of the impact of patching on league of legends. In *2016 International Conference on Collaboration Technologies and Systems (CTS)*, pages 128–135. IEEE, 2016.

[241] Nam Wook Kim, Benjamin Bach, Hyejin Im, Sasha Schriber, Markus Gross, and Hanspeter Pfister. Visualizing nonlinear narratives with story curves. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):595–604, 2018. doi: 10.1109/TVCG.2017.2744118.

[242] King. Candy crush saga. Mobile Game, 2012. URL `https://www.king.com/gam`

e/candycrush. Originally released on Facebook in April 2012; later released for iOS on November 14, 2012, and Android on December 14, 2012.

[243] Patrick Klepek. The small but important change "celeste" made to its celebrated assist mode. *VICE*, September 16 2019. URL https://www.vice.com/en/article/celeste-assist-mode-change-and-accessibility/.

[244] Kate Knibbs. Here's Proof You Can Train an AI Model Without Slurping Copyrighted Content — wired.com. https://www.wired.com/story/proof-you-can-train-ai-without-slurping-copyrighted-content/. [Accessed 07-06-2025].

[245] Athanasios Kokkinakis, Peter York, Moni Sagarika Patra, Justus Robertson, Ben Kirman, Alistair Coates, Alan Pedrassoli Pedrassoli Chitayat, Simon Demediuk, Anders Drachen, Jonathan Hook, et al. Metagaming and metagames in esports. *International Journal of Esports*, 1(1), 2021.

[246] Raph Koster. *Theory of fun for game design.* " O'Reilly Media, Inc.", 2013.

[247] Jason Krell. How does Riot Games balance 'League of Legends?' The answer is a bit meta. https://www.washingtonpost.com/video-games/esports/2021/02/22/league-legends-meta-riot/, 2021. [Accessed 13-09-2024].

[248] Max Kreminski. The dearth of the author in ai-supported writing. In *Proceedings of the Third Workshop on Intelligent and Interactive Writing Assistants*, pages 48–50, 2024.

[249] Max Kreminski, Isaac Karth, Michael Mateas, and Noah Wardrip-Fruin. Evaluating Mixed-Initiative Creative Interfaces via Expressive Range Coverage Analysis.

In *3rd Workshop on Human-AI Co-Creation with Generative Models*, volume 3124, March 2022.

[250] Max Kreminski, Isaac Karth, Michael Mateas, and Noah Wardrip-Fruin. Evaluating mixed-initiative creative interfaces via expressive range coverage analysis. In *IUI Workshops*, pages 34–45, 2022.

[251] K Yu Kristen, Matthew Guzdial, and Nathan R Sturtevant. Farmquest: A demonstration of an ai director video game test bed. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, pages 288–290, 2022.

[252] K Yu Kristen, Matthew Guzdial, and Nathan R Sturtevant. Evaluating the effects of ai directors for quest selection. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 20, pages 245–252, 2024.

[253] K Yu Kristen, Matthew Guzdial, and Nathan R Sturtevant. The farmquest player telemetry dataset: Playthrough data of a cozy farming game. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 20, pages 263–268, 2024.

[254] Jeppe Theiss Kristensen, Arturo Valdivia, and Paolo Burelli. Statistical modelling of level difficulty in puzzle games. In *2021 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2021.

[255] Fedwa Laamarti, Mohamad Eid, and Abdulmotaleb El Saddik. An overview of

serious games. *International Journal of Computer Games Technology*, 2014(1): 358152, 2014.

[256] Javier Laborda. Playgrounds as' workgrounds': How real-money trading transforms the gaming experience. In *Conference Proceedings of DiGRA 2024 Conference: Playgrounds*, 2024.

[257] Gorm Lai, William Latham, and Frederic Fol Leymarie. Towards Friendly Mixed Initiative Procedural Content Generation: Three Pillars of Industry, 2020. URL `https://arxiv.org/abs/2005.09324`. Version Number: 1.

[258] Gorm Lai, Frederic Fol Leymarie, and William Latham. On Mixed-Initiative Content Creation for Video Games. *IEEE Transactions on Games*, pages 1–1, 2022. ISSN 2475-1502, 2475-1510. doi: 10.1109/TG.2022.3176215. URL `https://ieeexplore.ieee.org/document/9779792/`.

[259] Patrick Lambe. Organising knowledge: taxonomies. *Knowledge and organisational effectiveness. Chandos, Oxford*, 2007.

[260] Petros Lameras, Sylvester Arnab, Ian Dunwell, Craig Stewart, Samantha Clarke, and Panagiotis Petridis. Essential features of serious games design in higher education: Linking learning attributes to game mechanics. *British journal of educational technology*, 48(4):972–994, 2017.

[261] Petri Lankoski. Embodiment in character-based videogames. In *Proceedings of the 20th international academic mindtrek conference*, pages 358–365, 2016.

[262] Chanel J Larche and Mike J Dixon. The relationship between the skill-challenge

balance, game expertise, flow and the urge to keep playing complex mobile games. *Journal of behavioral addictions*, 9(3):606–616, 2020.

[263] F Laudisa and C Rovelli. Stanford encyclopedia of philosophy. *See https://plato. stanford. edu/entries/qm-relational*, 2017.

[264] Effie L-C Law, Florian Brühlmann, and Elisa D Mekler. Systematic review and validation of the game experience questionnaire (geq)-implications for citation and reporting practice. In *Proceedings of the 2018 annual symposium on computer-human interaction in play*, pages 257–270, 2018.

[265] League of Legends. League balance for bronzies. `https://www.youtube.com/watch?v=BpBh9UMOvlU`, May 2021. YouTube video.

[266] Marc LeBlanc. Tools for creating dramatic game dynamics. *The game design reader: A rules of play anthology*, pages 438–459, 2006.

[267] Liel Leibovitz. *God in the machine: Video games as spiritual pursuit*. Templeton Foundation Press, 2014.

[268] Ryan Leigh, Justin Schonfeld, and Sushil J Louis. Using coevolution to understand and validate game balance in continuous games. In *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 1563–1570, 2008.

[269] Nuwan Leitan and Lucian Chaffey. Embodied cognition and its applications: A brief review. 2014.

[270] Levi H. S. Lelis, Willian M. P. Reis, and Ya'akov Gal. Procedural generation of

game maps with human-in-the-loop algorithms. *IEEE Transactions on Games*, 10(3):271–280, 2018. doi: 10.1109/TG.2017.2783361.

[271] Wilkins Leong, Julie Porteous, and John Thangarajah. Automated sifting of stories from simulated storyworlds. In *IJCAI*, pages 4950–4956, 2022.

[272] Ben Lewis-Evans. Throwing out the dopamine shots: Rewards without the neurotrash. Game Developers Conference, 2017.

[273] Danrui Li, Samuel S Sohn, Sen Zhang, Che-Jui Chang, and Mubbasir Kapadia. From words to worlds: Transforming one-line prompts into multi-modal digital stories with llm agents. In *Proceedings of the 17th ACM SIGGRAPH Conference on Motion, Interaction, and Games*, pages 1–12, 2024.

[274] Haodong Li, Gelei Deng, Yi Liu, Kailong Wang, Yuekang Li, Tianwei Zhang, Yang Liu, Guoai Xu, Guosheng Xu, and Haoyu Wang. Digger: Detecting copyright content mis-usage in large language model training, 2024. URL `https://arxiv.org/abs/2401.00676`.

[275] Xiaoxu Li, Yi Xia, Mustafa Can Gursesli, Xiao You, Siyuan Chen, and Ruck Thawonmas. Enhancing player experience in a first-person shooter with dynamic audio cue adjustment based on gaussian progress regression. *Applied Sciences*, 14 (23):11146, 2024.

[276] Nicholas Liao, Matthew Guzdial, and Mark Riedl. Deep convolutional player modeling on log and level data. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–4, 2017.

[277] Antonios Liapis, Georgios N Yannakakis, and Julian Togelius. Sentient sketchbook: computer-assisted game level authoring. 2013.

[278] Antonios Liapis, Gillian Smith, and Noor Shaker. Mixed-initiative content creation. *Procedural content generation in games*, pages 195–214, 2016.

[279] Antonios Liapis, Georgios N Yannakakis, Mark J Nelson, Mike Preuss, and Rafael Bidarra. Orchestrating game generation. *IEEE Transactions on Games*, 11(1): 48–68, 2018.

[280] Holin Lin and Chuen-Tsai Sun. Cash trade in free-to-play online games. *Games and Culture*, 6(3):270–287, 2011.

[281] Zeming Lin, Jonas Gehring, Vasil Khalidov, and Gabriel Synnaeve. Stardata: A starcraft ai research dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 13, pages 50–56, 2017.

[282] Conor Linehan, George Bellord, Ben Kirman, Zachary H Morford, and Bryan Roche. Learning curves: analysing pace and challenge in four successful puzzle games. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 181–190, 2014.

[283] Albert Liu and Steve Marschner. Balancing zero-sum games with one variable per strategy. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 13, pages 57–65, 2017.

[284] Jialin Liu, Julian Togelius, Diego Pérez-Liébana, and Simon M Lucas. Evolving

game skill-depth using general video game ai agents. In *2017 IEEE Congress on Evolutionary Computation (CEC)*, pages 2299–2307. IEEE, 2017.

[285] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *arXiv preprint arXiv:2307.03172*, 2023.

[286] Yun-En Liu, Erik Andersen, Richard Snider, Seth Cooper, and Zoran Popović. Feature-based projections for effective playtrace analysis. In *Proceedings of the 6th international conference on foundations of digital games*, pages 69–76, 2011.

[287] Jens-Martin Loebel and Jochen Koubek. Balancing act: Leveraging 'parameters'serious game as a tool for mastering game design in higher education. In *2024 IEEE Gaming, Entertainment, and Media Conference (GEM)*, pages 1–4. IEEE, 2024.

[288] Rafael Prieto De Lope and Nuria Medina-Medina. A comprehensive taxonomy for serious games. *Journal of Educational Computing Research*, 55(5):629–672, 2017. doi: 10.1177/0735633116681301. URL `https://doi.org/10.1177/0735633116681301`.

[289] Ricardo Lopes, Ken Hilf, Luke Jayapalan, and Rafael Bidarra. Mobile adaptive procedural content generation. In *Proceedings of the fourth workshop on Procedural Content Generation in Games (PCG 2013)*, Chania, Crete, Greece, 2013.

[290] Ludosity, Fair Play Labs. Nickelodeon all-star brawl, 2021.

[291] Jeremy Ludwig and Art Farley. A learning infrastructure for improving agent performance and game balance. *Proceedings of the AIIDE*, 7:7–12, 2007.

[292] Daniel MacCormick and Loutfouz Zaman. Echo: Analyzing gameplay sessions by reconstructing them from recorded data. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, pages 281–293, 2020.

[293] Keza MacDonald. The game design secrets of elden ring's hidetaka miyazaki. *The Guardian*, Jun 2024. URL `https://www.theguardian.com/games/article/2024/jun/26/pushing-buttons-meeting-hidetaka-miyazaki`.

[294] Maddy Makes Games. Celeste. [Linux, macOS, Nintendo Switch, Playstation 4, Windows, Xbox One, Google Stadia], 2018.

[295] Luís Fernando Maia, Windson Viana, and Fernando Trinta. Using monte carlo tree search and google maps to improve game balancing in location-based games. In *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 215–222. IEEE, 2017.

[296] Alessandro Maisto. Collaborative storytelling and llm: A linguistic analysis of automatically-generated role-playing game sessions, 2025. URL `https://arxiv.org/abs/2503.20623`.

[297] Solange Margarido, Penousal Machado, Licínio Roque, and Pedro Martins. Boosting mixed-initiative co-creativity in game design: A tutorial. *ACM Computing Surveys*, 2024.

[298] Alessandro Marincioni, Myriana Miltiadous, Katerina Zacharia, Rick Heemskerk,

Georgios Doukeris, Mike Preuss, and Giulio Barbero. The effect of llm-based npc emotional states on player emotions: An analysis of interactive game play. In *2024 IEEE Conference on Games (CoG)*, pages 1–6. IEEE, 2024.

[299] Julian Mariño, Willian Reis, and Levi Lelis. An empirical evaluation of evaluation metrics of procedurally generated mario levels. November 2015.

[300] Léa Martinez, Manuel Gimenes, and Eric Lambert. Entertainment video games for academic learning: A systematic review. *Journal of Educational Computing Research*, 60(5):1083–1109, 2022.

[301] Michael Mateas and Andrew Stern. Façade: An experiment in building a fully-realized interactive drama. In *Game developers conference*, volume 2, pages 4–8. Citeseer, 2003.

[302] Michael Mateas and Andrew Stern. Procedural authorship: A case-study of the interactive drama façade. *Digital Arts and Culture (DAC)*, 61, 2005.

[303] Michael Mateas and Andrew Stern. Structuring content in the façade interactive drama architecture. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 1, pages 93–98, 2005.

[304] Natalia Maury-Castañeda, Sergio Villarruel-Vasquez, and Willy Ugarte. Mas4games: A reinforced learning-based multi-agent system to improve player retention in virtual reality video games. In *International Conference on Computer-Human Interaction Research and Applications*, pages 104–120. Springer, 2023.

[305] Edwin McRae. What's the difference between procedural narrative and emergent

narrative? URL https://www.edmcrae.com/article/whats-the-difference-between-procedural-narrative-and-emergent-narrative#:~:text=Procedural%20Narrative%20%3D%20A%20design%20technique,that%20collection%20and%20interpretation%20process.

[306] Mega Crit. Slay the spire. Steam, GOG, PlayStation 4, Xbox One, Nintendo Switch, iOS, Android, Jan 2019. URL https://www.megacrit.com/. Video game.

[307] Sid Meier. Sid meier's civilization (video game series), 1991–2025. URL https://civilization.2k.com/. Turn-based strategy game franchise.

[308] Sid Meier. Sid meier's interesting decisions. https://www.youtube.com/watch?v=WggIdtrqgKg, Sep 2018. GDC Festival of Gaming, YouTube video.

[309] Marie Mejerwall. Growing an ai director into a full adventure director, March 2025.

[310] Luis Mejia-Puig and Tilanka Chandrasekera. The virtual body in a design exercise: a conceptual framework for embodied cognition. *International Journal of Technology and Design Education*, 33(5):1861–1882, 2023.

[311] Edward F Melcer. *Learning with the body: understanding the design space of embodied educational technology.* PhD thesis, New York University Tandon School of Engineering, 2018.

[312] Edward F Melcer and Marjorie Ann M Cuerdo. Death and rebirth in platformer games. *Game User Experience And Player-Centered Design*, pages 265–293, 2020.

[313] Edward F Melcer, Katelyn M Grasse, James Ryan, Nick Junius, Max Kreminski, Dietrich Squinkifer, Brent Hill, and Noah Wardrip-Fruin. Getting academical: a choice-based interactive storytelling game for teaching responsible conduct of research. In *Proceedings of the 15th International Conference on the Foundations of Digital Games*, pages 1–12, 2020.

[314] Brian Merchant. Ai is already taking jobs in the video game industry. URL `https://www.wired.com/story/ai-is-already-taking-jobs-in-the-video-game-industry/`.

[315] Ailea Merriam-Pigg. Metagaming attention: Defining the metagame through the economy of attention on twitch. *Popular Culture Studies Journal*, 9(2):55–68, 2022.

[316] Panagiotis Migkotzidis and Antonios Liapis. Susketch: Surrogate models of gameplay as a design assistant. *IEEE Transactions on Games*, 14(2):273–283, 2021.

[317] miHoYo. Genshin impact. [DIGITAL], 2020.

[318] Konstantin Mitgutsch and Narda Alvarado. Purposeful by design? a serious game design assessment framework. In *Proceedings of the International Conference on the foundations of digital games*, pages 121–128, 2012.

[319] Youichiro Miyake. Machine learning summit: 'saga emerald beyond': Balancing battle with reinforcement learning ai. Game Developers Conference (GDC) Vault, 2024. URL `https://gdcvault.com/play/1034795/Machine-Learning-Summit-SaGa-Emerald`. Presented at the Machine Learning Summit, GDC 2024.

[320] Rebecca Moden. Blank page syndrome and how to beat it i oxford open learning, October 2022. URL https://www.ool.co.uk/blog/blank-page-syndrome-and-how-to-beat-it/.

[321] Mojang. Minecraft. [DIGITAL], 2011.

[322] Jurike V Moniaga, Andry Chowanda, Agus Prima, M Dimas Tri Rizqi, et al. Facial expression recognition as dynamic game balancing system. *Procedia Computer Science*, 135:361–368, 2018.

[323] Mihail Morosan and Riccardo Poli. Automated game balancing in ms pacman and starcraft using evolutionary algorithms. In *Applications of Evolutionary Computation: 20th European Conference, EvoApplications 2017, Amsterdam, The Netherlands, April 19-21, 2017, Proceedings, Part I 20*, pages 377–392. Springer, 2017.

[324] Hannah Morrison and Chris Martens. A generative model of group conversation. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, pages 1–7, Hyannis Massachusetts, August 2017. ACM. ISBN 978-1-4503-5319-9. doi: 10.1145/3102071.3116218. URL https://dl.acm.org/doi/10.1145/3102071.3116218.

[325] Fatemeh Mortazavi, Hadi Moradi, and Abdol-Hossein Vahabie. Dynamic difficulty adjustment approaches in video games: a systematic literature review. *Multimedia Tools and Applications*, 83(35):83227–83274, 2024.

[326] The Mountaineers. *Mountaineering: The Freedom of the Hills*. Mountaineers Books, Seattle, WA, 9th edition, 2017. ISBN 978-1680510041.

[327] Talal Musa. "players will exceed our expectations": Bungie on the art of destiny 2's weapon balancing, Aug 2024. URL https://www.videogamer.com/news/players-will-exceed-our-expectations-bungie-on-the-art-of-destiny-2s-weapon-balancing/.

[328] Mahin Naderifar, Hamideh Goli, and Fereshteh Ghaljaie. Snowball sampling: A purposeful method of sampling in qualitative research. *Strides in development of medical education*, 14(3), 2017.

[329] Muhammad Naeem, Wilson Ozuem, Kerry Howell, and Silvia Ranfagni. A step-by-step process of thematic analysis to develop a conceptual model in qualitative research. *International Journal of Qualitative Methods*, 22:16094069231205789, 2023. doi: 10.1177/16094069231205789. URL https://doi.org/10.1177/16094069231205789.

[330] Thomas Nagel. *What is it like to be a bat?* Oxford University Press, 2024.

[331] Jeanne Nakamura, Mihaly Csikszentmihalyi, et al. Flow theory and research. *Handbook of positive psychology*, 195:206, 2009.

[332] Navapat Nananukul and Wichayaporn Wongkamjan. What if red can talk? dynamic dialogue generation using large language models. *arXiv preprint arXiv:2407.20382*, 2024.

[333] Tom Nardi. Video game preservation through decompilation, June 23 2025. URL https://hackaday.com/2025/06/23/video-game-preservation-through-decompilation/.

[334] Mark Nelson, Calvin Ashmore, and Michael Mateas. Authoring an interactive narrative with declarative optimization-based drama management. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 2, pages 127–129, 2006.

[335] NEON AURELIUS. Sparrow warfare, 2026. URL `https://store.steampowered.com/app/3473440/Sparrow_Warfare/`.

[336] Robert C Nickerson, Upkar Varshney, and Jan Muntermann. A method for taxonomy development and its application in information systems. *European Journal of Information Systems*, 22(3):336–359, 2013.

[337] Ed Nightingale. Players have translated tunic's runic language. *Eurogamer*, April 2022. URL `https://www.eurogamer.net/players-have-translated-tunics-runic-language`.

[338] Sofia Maria Nikolakaki, Ogheneovo Dibie, Ahmad Beirami, Nicholas Peterson, Navid Aghdaie, and Kazi Zaman. Competitive balance in team sports games. In *2020 IEEE Conference on Games (CoG)*, pages 526–533. IEEE, 2020.

[339] Nintendo Co. Ltd. Super smash bros., 1999.

[340] Nintendo EPD. The legend of zelda: breath of the wild. [Nintendo Switch, Wii U, Nintendo Switch 2], 2017.

[341] Nintendo EPD. Animal crossing: new horizons. [DIGITAL], 2020.

[342] Ashey Noblega, Aline Paes, and Esteban Clua. Towards adaptive deep reinforcement game balancing. In *ICAART (2)*, pages 693–700, 2019.

[343] Donald A Norman. *Emotional Design: Why We Love (or Hate) Everyday Things*. Basic Books, 2004.

[344] A Norman Donald. *The design of everyday things*. MIT Press, 2013.

[345] Anna Maria Oberländer, Benedict Lösser, and Daniel Rau. Taxonomy research in information systems: A systematic assessment. 2019.

[346] Obsidian Entertainment. Fallout: new vegas. [DIGITAL], 2010.

[347] Adam Millard The Architect of Games. Why are games so hard to balance?, July 2018. URL `https://youtube.com/watch?v=K3n-Sy2Ko4I`. YouTube video.

[348] Jacob Kaae Olesen, Georgios N Yannakakis, and John Hallam. Real-time challenge balance in an rts game using rtneat. In *2008 IEEE symposium on computational intelligence and games*, pages 87–94. IEEE, 2008.

[349] Masahiro Sakurai on Creating Games. Give yourself a handicap when balancing your game [game essence], October 2024. URL `https://youtube.com/watch?v=2T86JqRAKjc`. YouTube video.

[350] Brian O'Neill and Mark Riedl. Toward a computational framework of suspense and dramatic arc. In Sidney D'Mello, Arthur Graesser, Björn Schuller, and Jean-Claude Martin, editors, *Affective Computing and Intelligent Interaction*, pages 246–255, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-24600-5.

[351] Origin Systems. Ultima. [Amiga], 1981.

[352] Joseph Osborn, Ben Samuel, Joshua McCoy, and Michael Mateas. Evaluating play trace (dis) similarity metrics. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 10, pages 139–145, 2014.

[353] Joseph Osborn, Ben Samuel, Michael Mateas, and Noah Wardrip-Fruin. Playspecs: Regular expressions for game play traces. In *Proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment*, volume 11, pages 170–176, 2015.

[354] Jose Otero. Super smash bros. mod Project M shut down. *IGN*, December 2015. URL `https://www.ign.com/articles/2015/12/02/super-smash-bros-mod-project-m-shut-down`. Details the sudden cancellation of the mod and its impact on the competitive community.

[355] Bobby Owsinski. *The Mixing Engineer's Handbook*. Bobby Owsinski Media Group, Burbank, CA, 4th edition, 2017. ISBN 978-0988839182.

[356] Kristine Oygardslia, Charlotte Weitze, and Jh Shin. The educational potential of visual novel games: Principles for design. *Replaying Japan*, 2, 03 2020.

[357] Lucy O'Brien. How ubisoft's new generative ai prototype changes the narrative for npcs, March 2024. URL `https://news.ubisoft.com/en-us/article/5qXdxhshJBXoanFZApdG3L`.

[358] DJ O'Connor. Emergent properties. In *Old and new questions in physics, cosmol-*

ogy, philosophy, and theoretical biology: Essays in honor of Wolfgang Yourgrau, pages 719–732. Springer, 2015.

[359] Casey O'Donnell. The everyday lives of video game developers: Experimentally understanding underlying systems/structures. *Transformative Works and Cultures*, 2(1):1–11, 2009.

[360] Randy Pagulayan, Kevin Keeker, Dennis Wixon, Ramon Romero, and Thomas Fuller. User-centered design in games. *Human-Computer Interact. Handb*, pages 883–906, 01 2003. doi: 10.1201/b11963-39.

[361] Nathan Partlan, Elin Carstensdottir, Erica Kleinman, Sam Snodgrass, Casper Harteveld, Gillian Smith, Camillia Matuk, Steven C Sutherland, and Magy Seif El-Nasr. Evaluation of an automatically-constructed graph-based representation for interactive narrative. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, pages 1–9, 2019.

[362] Christopher Pedersen, Julian Togelius, and Georgios N Yannakakis. Modeling player experience for content creation. *IEEE Transactions on Computational Intelligence and AI in Games*, 2(1):54–67, 2010.

[363] Caroline Pelletier. Charming work: An ethnography of an indie game studio. *Convergence*, 28(2):561–578, 2022.

[364] Sumedh Pendurkar and Chris Chow. Bilevel entropy based mechanism design for balancing meta in video games. In *Proceedings of the 2023 International Con-*

ference on Autonomous Agents and Multiagent Systems. Proceedings of the 2023 International Conference on Autonomous Agents and ..., 2023.

[365] Xiangyu Peng, Jessica Quaye, Sudha Rao, Weijia Xu, Portia Botchway, Chris Brockett, Nebojsa Jojic, Gabriel DesGarennes, Ken Lobb, Michael Xu, et al. Player-driven emergence in llm-driven game narrative. In *2024 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2024.

[366] People Make Games. Why the sound of a gun had to be nerfed in wolfenstein: Enemy territory, August 29 2019. URL `https://www.youtube.com/watch?v=RDxiuHdR_T4`. YouTube video, 12:19.

[367] Diego Perez-Liebana, Jialin Liu, Ahmed Khalifa, Raluca D Gaina, Julian Togelius, and Simon M Lucas. General video game ai: A multitrack framework for evaluating agents, games, and content generation algorithms. *IEEE Transactions on Games*, 11(3):195–214, 2019.

[368] Daniel Perez-Marcos. Virtual reality experiences, embodiment, videogames and their dimensions in neurorehabilitation. *Journal of neuroengineering and rehabilitation*, 15(1):113, 2018.

[369] Johannes Pfau and Magy Seif El-Nasr. On video game balancing: Joining player- and data-driven analytics. *ACM Games: Research and Practice*, 2024.

[370] Johannes Pfau, Antonios Liapis, Georg Volkmar, Georgios N Yannakakis, and Rainer Malaka. Dungeons & replicants: automated game balancing via deep

player behavior modeling. In *2020 IEEE Conference on Games (CoG)*, pages 431–438. IEEE, 2020.

[371] Johannes Pfau, Antonios Liapis, Georgios N Yannakakis, and Rainer Malaka. Dungeons & replicants ii: automated game balancing across multiple difficulty dimensions via deep player behavior modeling. *IEEE Transactions on Games*, 15 (2):217–227, 2022.

[372] PG Stats. List of SSBM tier lists (NTSC) - SmashWiki, the Super Smash Bros. wiki. SSBWiki, 2021. URL `https://www.ssbwiki.com/List_of_SSBM_tier_lists_(NTSC)`.

[373] Martin Pichlmair and Mads Johansen. Designing game feel: A survey. *IEEE Transactions on Games*, 14(2):138–152, 2021.

[374] Catherine Plaisant, David Carr, and Ben Shneiderman. Image-browser taxonomy and guidelines for designers. *Ieee Software*, 12(2):21–32, 1995.

[375] Luke Plunkett. "an overwhelmingly negative and demoralizing force": What it's like working for a company that's forcing ai on its developers, Apr 2025. URL `https://aftermath.site/ai-video-game-development-art-vibe-coding-midjourney`.

[376] Lev Poretski and Ofer Arazy. Placing value on community co-creations: A study of a video game'modding'community. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing*, pages 480–491, 2017.

[377] Anna Samira Praetorius and Daniel Görlich. How avatars influence user behavior: A review on the proteus effect in virtual environments and video games. In *Proceedings of the 15th International Conference on the Foundations of Digital Games*, pages 1–9, 2020.

[378] Mike Preuss, Thomas Pfeiffer, Vanessa Volz, and Nicolas Pflanzl. Integrated balancing of an rts game: Case study and toolbox refinement. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2018.

[379] Vahid Ranandeh and Pejman Mirza-Babaei. Beyond equilibrium: Utilizing ai agents in video game economy balancing. In *Companion Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, pages 155–160, 2023.

[380] Harvey Randall. Doom (2016)'s creative director reveals the secret to balancing the iconic BFG during a major charity speedrun: 'Uh, we don't' — pcgamer.com. `https://www.pcgamer.com/doom-2016s-creative-director-reveals-the-secret-to-balancing-the-iconic-bfg-during-a-major-charity-speedrun-uh-we-dont/`, 2024. [Accessed 13-09-2024].

[381] Harvey Randall. Fallout: New Vegas director reveals that game balance is 'mostly vibes based', says he only used a weapons spreadsheet 'for maybe a couple of months' — pcgamer.com. `https://www.pcgamer.com/games/fallout/fallout-new-vegas-director-reveals-that-game-balance-is-mostly-vibes-based-says-he-only-used-a-weapons-spreadsheet-for-maybe-a-couple-of-months/`, 2024. [Accessed 13-09-2024].

[382] Nintendo R&D1. Metroid, 1986. URL https://en.wikipedia.org/wiki/Metroid_(game).

[383] Andrew J Reagan, Lewis Mitchell, Dilan Kiley, Christopher M Danforth, and Peter Sheridan Dodds. The emotional arcs of stories are dominated by six basic shapes. *EPJ data science*, 5(1):1–12, 2016.

[384] Reddit Inc. Reddit r/gamedev search: "game balance". https://www.reddit.com/r/gamedev/search/?q=%22game+balance%22, 2026. Internal community search for "game balance" within r/gamedev. [Accessed 07-02-2026].

[385] Jawad Jandali Refai, Scott Bateman, and Michael W. Fleming. External assistance techniques that target core game tasks for balancing game difficulty. *Frontiers in Computer Science*, Volume 2 - 2020, 2020. ISSN 2624-9898. doi: 10.3389/fcomp.2020.00017. URL https://www.frontiersin.org/journals/computer-science/articles/10.3389/fcomp.2020.00017.

[386] Simão Reis, Luís Paulo Reis, and Nuno Lau. Automatic generation of a suboptimal agent population with learning. In *New Knowledge in Information Systems and Technologies: Volume 2*, pages 65–74. Springer, 2019.

[387] Simão Reis, Luís Paulo Reis, and Nuno Lau. Player engagement enhancement with video games. In *New Knowledge in Information Systems and Technologies: Volume 2*, pages 263–272. Springer, 2019.

[388] Simão Reis, Luís Paulo Reis, and Nuno Lau. Game adaptation by using reinforce-

ment learning over meta games. *Group Decision and Negotiation*, 30(2):321–340, 2021.

[389] Simão Reis, Luís Paulo Reis, and Nuno Lau. Vgc ai competition-a new model of meta-game balance ai competition. In *2021 IEEE Conference on Games (CoG)*, pages 01–08. IEEE, 2021.

[390] Simão Reis, Rita Novais, Luís Paulo Reis, and Nuno Lau. An adversarial approach for automated pokémon team building and meta-game balance. *IEEE Transactions on Games*, 2023.

[391] Simão Reis, Rita Novais, Luís Paulo Reis, and Nuno Lau. Automatic difficulty balance in two-player games with deep reinforcement learning. In *2023 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2023.

[392] Alex Reisner. The unbelievable scale of ai's pirated-books problem, Mar 2025. URL `https://www.theatlantic.com/technology/archive/2025/03/libgen-meta-openai/682093/`.

[393] Judith S Reitman. Skilled perception in go: Deducing memory structures from inter-response times. *Cognitive psychology*, 8(3):336–356, 1976.

[394] Shaolei Ren, Bill Tomlinson, Rebecca W Black, and Andrew W Torrance. Reconciling the contrasting narratives on the environmental impact of large language models. *Scientific Reports*, 14(1):26310, 2024.

[395] Richard Garfield. Magic: the gathering. [Physical], 1993.

[396] Mark O Riedl and Andrew Stern. Failing believably: Toward drama management with autonomous actors in interactive narratives. In *International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, pages 195–206. Springer, 2006.

[397] Mark Owen Riedl and Vadim Bulitko. Interactive narrative: An intelligent systems approach. *Ai Magazine*, 34(1):67–67, 2013.

[398] Alex Rietveld, Sander Bakkes, and Diederik Roijers. Circuit-adaptive challenge balancing in racing games. In *2014 IEEE Games Media Entertainment*, pages 1–8. IEEE, 2014.

[399] Riot GalaxySmash. Quick gameplay thoughts 4/22. `https://www.leagueoflegends.com/en-us/news/dev/quick-gameplay-thoughts-4-22/`, Apr 2022. League of Legends Dev Blog.

[400] Riot Games. League of legends. [DIGITAL], 2009.

[401] Horst WJ Rittel and Melvin M Webber. Dilemmas in a general theory of planning. *Policy Sciences*, 4(2):155–169, 1973.

[402] Warren Robinett. Synthetic experience: A proposed taxonomy. *Presence: Teleoperators & Virtual Environments*, 1(2):229–247, 1992.

[403] Katja Rogers, Mark Colley, David Lehr, Julian Frommel, Marcel Walch, Lennart E Nacke, and Michael Weber. Kickar: Exploring game balancing through boosts and handicaps in augmented reality table football. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2018.

[404] Sonny Rosenthal and Rabindra A Ratan. Balancing learning and enjoyment in serious games: Kerbal space program and the communication mediation model. *Computers & Education*, 182:104480, 2022.

[405] William David Ross et al. *Aristotle's metaphysics*, volume 2. Clarendon Press, 1924.

[406] John A Rothchild and Daniel Rothchild. Copyright implications of the use of code repositories to train a machine learning model. *Free Software Foundation*, 2022.

[407] Florian Rupp and Kai Eckert. Geevo: Game economy generation and balancing with evolutionary algorithms. *arXiv preprint arXiv:2404.18574*, 2024.

[408] Florian Rupp and Kai Eckert. Level the level: Balancing game levels for asymmetric player archetypes with reinforcement learning. In *Proceedings of the 20th International Conference on the Foundations of Digital Games*, pages 1–4, 2025.

[409] Florian Rupp, Manuel Eberhardinger, and Kai Eckert. Simulation-driven balancing of competitive game levels with reinforcement learning. *IEEE Transactions on Games*, 2024.

[410] Florian Rupp, Alessandro Puddu, Christian Becker-Asano, and Kai Eckert. It might be balanced, but is it actually good? an empirical evaluation of game level balancing. In *2024 IEEE Conference on Games (CoG)*, pages 1–4. IEEE, 2024.

[411] Masahiro Sakurai. Using parameters to establish characters [planning & game design], April 9 2024. URL `https://www.youtube.com/watch?v=zwiS1L6QVY0`. Masahiro Sakurai on Creating Games.

[412] Masahiro Sakurai. Average and mediocre are the same thing [design specifics], May 21 2024. URL https://www.youtube.com/watch?v=KW4x-_Y8BmM. Masahiro Sakurai on Creating Games.

[413] J. Saldana. *The Coding Manual for Qualitative Researchers*. SAGE Publications, 2012. ISBN 9781446271421. URL https://books.google.com/books?id=V3tTG4jvgFkC.

[414] Joni Salminen, Soon-Gyo Jung, Jukka Vahlo, Shammur Chowdhury, Aki Koponen, and Bernard James Jansen. Developing prototype player personas from game preferences. In *CHI'20: CHI Conference on Human Factors in Computing Systems*, 2018.

[415] Joni Salminen, Kathleen Guan, Soon-gyo Jung, Shammur A Chowdhury, and Bernard J Jansen. A literature review of quantitative persona creation. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pages 1–14, 2020.

[416] Joni Salminen, Kathleen Wenyun Guan, Soon-Gyo Jung, and Bernard Jansen. Use cases for design personas: A systematic review and new frontiers. pages 1–21. ACM, 4 2022. ISBN 9781450391573. doi: 10.1145/3491102.3517589. URL https://dl.acm.org/doi/10.1145/3491102.3517589.

[417] Sandfall Interactive. Clair obscur: expedition 33. [PlayStation 5, Windows, Xbox Series X/S], 2025.

[418] Daiki Satoi and Yuta Mizuno. Meta artificial intelligence and artificial intelligence

director. In *Encyclopedia of Computer Graphics and Games*, pages 1119–1126. Springer, 2024.

[419] Tom Schaul. A video game description language for model-based or interactive learning. In *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, pages 1–8. IEEE, 2013.

[420] Jesse Schell. *The Art of Game Design: A book of lenses*. CRC press, 2008.

[421] Matthew Schomer. Sakurai Reveals Surprising Super Smash Bros. Ultimate Stat — gamerant.com. `https://gamerant.com/super-smash-bros-ultimate-character-win-rates/`, 2024. [Accessed 13-09-2024].

[422] Ian Schreiber and Brenda Romero. *Game balance*. CRC Press, 2021.

[423] Natasha Dow Schüll. Addiction by design: Machine gambling in las vegas. In *Addiction by design*. Princeton university press, 2012.

[424] Cliff Scott and Melissa Medaugh. Axial coding. *The international encyclopedia of communication research methods*, 10:9781118901731, 2017.

[425] Seeking Alpha. Ubisoft entertainment's (ubsff) ceo yves guillemot on q4 2017 results: Earnings call transcript. `https://web.archive.org/web/20250821195324/https://seekingalpha.com/article/4073873-ubisoft-entertainments-ubsff-ceo-yves-guillemot-on-q4-2017-results-earnings-call-transcript`, May 2017. Archived via Wayback Machine; Accessed: February 8, 2026.

[426] Caetano Vieira Neto Segundo, K Emerson, A Calixto, and R Gusmao. Dynamic

difficulty adjustment through parameter manipulation for space shooter game. *Proceedings of SBGames*, pages 234–237, 2016.

[427] Michael Sellers. *Advanced game design: a systems approach*. Addison-Wesley Professional, 2017.

[428] Sabrina A Sgandurra. Fight. heal. repeat: A look at rhetorical devices in grinding game mechanics. *Simulation & Gaming*, 53(4):388–399, 2022.

[429] Noor Shaker, Georgios Yannakakis, and Julian Togelius. Towards automatic personalized content generation for platform games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 6, pages 63–68, 2010.

[430] Noor Shaker, Julian Togelius, and Mark J Nelson. *Procedural content generation in games*. Springer, 2016.

[431] Manu Sharma, Santiago Ontañón, Manish Mehta, and Ashwin Ram. Drama management and player modeling for interactive fiction games. *Computational Intelligence*, 26(2):183–211, 2010.

[432] Samuel Shields and Edward F. Melcer. FighterDDA: A simulation testbed for evaluating director-based dynamic balancing. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*. AAAI Press, November 2025. In press.

[433] Samuel Shields, Ross Mawhorter, Edward Melcer, and Michael Mateas. Searching for balanced 2d brawler games: successes and failures of automated evaluation.

In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 18, pages 189–198, 2022.

[434] Samuel Shields, Alexander Calderwood, Shi Johnson-Bey, Noah Wardrip-Fruin, Michael Mateas, and Edward Melcer. Generating together: Lessons learned from developing an educational visual novel with ai collaboration. In *2024 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2024.

[435] Samuel Shields, Michael Mateas, and Edward Melcer. Towards qualia-driven design. In *Conference Proceedings of DiGRA 2024 Conference: Playgrounds*, 2024.

[436] Samuel Shields, Noah Wardrip-Fruin, and Edward F. Melcer. Playtrace arc search: A tool to explore and evaluate large spaces of playtrace metrics through user-defined curves. In *Proceedings of the 12th Experimental Artificial Intelligence in Games (EXAG 2025) Workshop co-located with the 21st AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2025), Edmonton, Alberta, Canada, November 2025*, 2025. In press.

[437] Samuel Shields, Noah Wardrip-Fruin, and Edward F Melcer. Searching playtrace data to identify and evaluate dramatic arcs in game systems. In *International Conference on Interactive Digital Storytelling*, pages 385–395. Springer, 2025.

[438] Samuel Shields, Oliver Withington, and Edward F. Melcer. Designer difficulties: Visualizing the possibility spaces of dynamic difficulty adjustment systems. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*. AAAI Press, November 2025. In press.

[439] Kyong Jin Shim and Jaideep Srivastava. Behavioral profiles of character types in everquest ii. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, pages 186–194. IEEE, 2010.

[440] Emily Short. Analysis: Conversation design in games, May 2009. URL `https://www.gamedeveloper.com/game-platforms/analysis-conversation-design-in-games`.

[441] Tanya X Short and Tarn Adams. *Procedural storytelling in game design*. Crc Press, 2019.

[442] Miguel Sicart. Playthings. *Games and culture*, 17(1):140–155, 2022.

[443] Md Abu Bakar Siddik, Maria Amaya, and Landon T Marston. The water and carbon footprint of cryptocurrencies and conventional currencies. *Journal of Cleaner Production*, 411:137268, 2023.

[444] Sierra Studios. Homeworld. [Windows, OS X], 1999.

[445] Rafet Sifa, Anders Drachen, and Christian Bauckhage. Profiling in games: Understanding behavior from telemetry. *Social interactions in virtual worlds: An interdisciplinary perspective*, pages 337–375, 2018.

[446] Inês Silva, Pedro Cardoso, and Eva Oliveira. Narrative and gameplay: the balanced and imbalanced relationship between dramatic tension and gameplay tension. In *Proceedings of the 9th International Conference on Digital and Interactive Arts*, pages 1–8, 2019.

[447] Mirna Paula Silva, Victor do Nascimento Silva, and Luiz Chaimowicz. Dynamic difficulty adjustment on moba games. *Entertainment Computing*, 18:103–123, 2017.

[448] Eirik H Skjærseth and Harald Vinje. Evolutionary algorithms for generating interesting fighting game character mechanics. Master's thesis, NTNU, 2020.

[449] Adam Smith, Mark Nelson, and Michael Mateas. Prototyping games with biped. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 5, pages 193–194, 2009.

[450] Adam M Smith, Mark J Nelson, and Michael Mateas. Ludocore: A logical game engine for modeling videogames. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, pages 91–98. IEEE, 2010.

[451] Adam M Smith, Chris Lewis, Kenneth Hullet, Gillian Smith, and Anne Sullivan. An inclusive view of player modeling. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, pages 301–303, 2011.

[452] Adam Marshall Smith. *Mechanizing exploratory game design*. PhD thesis, University of California, Santa Cruz, 2012.

[453] David Woodruff Smith. Phenomenology. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2023 edition, 2023.

[454] Gillian Smith. Procedural content generation: An overview. *Level Design*, pages 159–183, 2017.

[455] Gillian Smith and Jim Whitehead. Analyzing the expressive range of a level generator. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games - PCGames '10*, pages 1–7, Monterey, California, 2010. ACM Press. ISBN 978-1-4503-0023-0. doi: 10.1145/1814256.1814260. URL http://portal.acm.org/citation.cfm?doid=1814256.1814260.

[456] Gillian Smith, Mike Treanor, Jim Whitehead, and Michael Mateas. Rhythm-based level generation for 2d platformers. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, FDG '09, page 175–182, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584379. doi: 10.1145/1536513.1536548. URL https://doi.org/10.1145/1536513.1536548.

[457] Sora Ltd. Super smash bros. brawl, 2008.

[458] Kynan Sorochan, Jerry Chen, Yakun Yu, and Matthew Guzdial. Generating lode runner levels by learning player paths with lstms. In *The 16th International Conference on the Foundations of Digital Games (FDG) 2021*, pages 1–7, 2021.

[459] Square. Final fantasy. [Nintendo Entertainment System], 1987.

[460] Square. Final fantasy vii. [PlayStation], 1997.

[461] Square Enix. Saga emerald beyond. Video Game, 2024. URL https://saga-franchise.square-enix-games.com/en-us/games/sgeb. Available on Nintendo Switch, PlayStation 5, PlayStation 4, PC, iOS, and Android.

[462] Christopher Dristig Stenström and Staffan Björk. Understanding computer role-

playing games: A genre analysis based on gameplay features in combat systems. In *Second Workshop on Design Patterns in Games (FDG 2013)*, 2013.

[463] Mike Stout. Enemy Attacks and Telegraphing — gamedeveloper.com. `https://www.gamedeveloper.com/design/enemy-attacks-and-telegraphing`, 2015. [Accessed 12-06-2025].

[464] David Sudnow. *Pilgrim in the Microworld*. Warner Books New York, 1983.

[465] Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*, 10(3):257–270, 2018.

[466] Summoner's Rift Team. Balance blog collection. `https://www.leagueoflegends.com/en-us/news/dev/balance-blog-collection/`, May 2021. League of Legends Dev Blog.

[467] Yuqian Sun, Zhouyi Li, Ke Fang, Chang Hee Lee, and Ali Asadipour. Language as reality: a co-creative storytelling game experience in 1001 nights using generative ai. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 19, pages 425–434, 2023.

[468] Supergiant. Hades 2. [DIGITAL], 2024.

[469] Clifford J Sussman, James M Harper, Jessica L Stahl, and Paul Weigle. Internet and video game addictions: Diagnosis, epidemiology, and neurobiology. *Child and Adolescent Psychiatric Clinics*, 27(2):307–326, 2018.

[470] Jan Švelch. Resisting the perpetual update: Struggles against protocological power in video games. *New Media & Society*, 21(7):1594–1612, 2019.

[471] Penelope Sweetser. Emergence in games. *Game Connect Asia Pacific 2008*, 2008.

[472] Penelope Sweetser and Peta Wyeth. Gameflow: a model for evaluating player enjoyment in games. *Computers in Entertainment (CIE)*, 3(3):3–3, 2005.

[473] Maciej Swiechowski. Featured Blog — Supporting game design with evolutionary algorithms — gamedeveloper.com. `https://www.gamedeveloper.com/design/supporting-game-design-with-evolutionary-algorithms`, 2024. [Accessed 13-09-2024].

[474] Steve Swink. *Game feel: a game designer's guide to virtual sensation*. CRC press, 2008.

[475] JGG Taelman. Playing with the script: Super smash bros. melee. from a casual game to a competitive game. Master's thesis, 2015.

[476] thatgamecompany. Journey. [Playstation 3, Playstation 4, Windows, iOS], 2012.

[477] Tommy Thompson. AI 101: Introducing Utility AI — aiandgames.com. `https://www.aiandgames.com/p/ai-101-introducing-utility-ai`, 2025. [Accessed 31-05-2025].

[478] Anne Mette Thorhauge. The steam platform economy: From retail to player-driven economies. *new media & society*, 26(4):1963–1983, 2024.

[479] David Thue and Vadim Bulitko. Toward a unified understanding of experience

management. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 14, pages 130–136, 2018.

[480] David Thue, Vadim Bulitko, and Marcia Spetch Spetch. Passage: A demonstration of player modelling in interactive storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 4, pages 226–227, 2008.

[481] Tim JW Tijs, Dirk Brokken, and Wijnand A IJsselsteijn. Dynamic game balancing by recognizing affect. In *Fun and Games: Second International Conference, Eindhoven, The Netherlands, October 20-21, 2008. Proceedings*, pages 88–93. Springer, 2008.

[482] Julian Togelius and Jurgen Schmidhuber. An experiment in automatic game design. In *2008 IEEE Symposium On Computational Intelligence and Games*, pages 111–118. Citeseer, 2008.

[483] Julian Togelius and Noor Shaker. The search-based approach. In *Procedural Content Generation in Games*, pages 17–30. Springer, 2016.

[484] Julian Togelius, Georgios N Yannakakis, Kenneth O Stanley, and Cameron Browne. Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3):172–186, 2011.

[485] Weimin Toh. The player experience and design implications of narrative games. *International Journal of Human–Computer Interaction*, 39(13):2742–2769, 2023.

[486] Nenad Tomašev, Ulrich Paquet, Demis Hassabis, and Vladimir Kramnik. Assessing game balance with alphazero: Exploring alternative rule sets in chess. *arXiv preprint arXiv:2009.04374*, 2020.

[487] Gustavo F Tondello, Rina R Wehbe, Rita Orji, Giovanni Ribeiro, and Lennart E Nacke. A framework and taxonomy of videogame playing preferences. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*, pages 329–340, 2017.

[488] Game Maker's Toolkit. How games get balanced, April 2019. URL `https://youtube.com/watch?v=WXQzdXPTb2A`. YouTube video.

[489] Turndownforwalt. How did donkey kong win a melee tournament?, July 16 2024. URL `https://www.youtube.com/watch?v=VcNlSH0FrzQ`.

[490] Twine. URL `https://twinery.org/cookbook/storylets/harlowe/harlowe_storylets.html#:~:text=Storylets%20are%20a%20different%20way,often%20useful%20in%20understanding%20storylets`.

[491] Anders Tychsen and Alessandro Canossa. Defining personas in games using metrics. In *Proceedings of the 2008 conference on future play: Research, play, share*, pages 73–80, 2008.

[492] Michael Tye. Qualia. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Fall 2025 edition, 2025.

[493] Michael Unterkalmsteiner and Waleed Abdeen. A compendium and evaluation

of taxonomy quality attributes. *Expert Systems*, 40(1):e13098, 2023. doi: https://doi.org/10.1111/exsy.13098. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/exsy.13098.

[494] Jukka Vahlo. Player persona research. In *Abstract Proceedings of DiGRA 2019 Conference: Game, Play and the Emerging Ludo-Mix*, 2019.

[495] Rebekah Valentine. How datamining has changed the landscape of video game puzzle and secret design, August 22 2024. URL https://www.ign.com/articles/how-datamining-has-changed-the-landscape-of-video-game-puzzle-and-secret-design.

[496] Josep Valls-Vargas, Santiago Ontanón, and Jichen Zhu. Exploring player trace segmentation for dynamic play style prediction. In *Proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment*, volume 11, pages 93–99, 2015.

[497] Valve Corporation. Artifact. Video game [PC], 2018. URL https://playartifact.com/.

[498] Valve South. Left 4 dead. [Windows, Xbox 360, macOS], 2008.

[499] Bessel Van der Kolk. The body keeps the score: Brain, mind, and body in the healing of trauma. *New York*, 3:14–211, 2014.

[500] Giel Van Lankveld, Pieter Spronck, H Jaap Van Den Herik, and Matthias Rauterberg. Incongruity-based adaptive game balancing. In *Advances in Computer*

Games: 12th International Conference, ACG 2009, Pamplona Spain, May 11-13, 2009. Revised Papers 12, pages 208–220. Springer, 2010.

[501] Livia Veselka, Rochelle Wijesingha, Scott T Leatherdale, Nigel E Turner, and Tara Elton-Marshall. Factors associated with social casino gaming among adolescents across game types. *BMC Public Health*, 18(1):1167, 2018.

[502] Rodrigo Vicencio-Moreira, Regan L Mandryk, and Carl Gutwin. Balancing multiplayer first-person shooter games using aiming assistance. In *2014 IEEE Games Media Entertainment*, pages 1–8. IEEE, 2014.

[503] Rodrigo Vicencio-Moreira, Regan L Mandryk, and Carl Gutwin. Now you can compete with anyone: Balancing players of different skill levels in a first-person shooter game. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 2255–2264, 2015.

[504] Jennifer Vilcarino and Lauraine Langreo. Rising use of ai in schools comes with big downsides for students. *Education Week*, 2025.

[505] Eugene Volokh. Copyright office: No copyright protection for certain ai-generated works, February 2023. URL `https://reason.com/volokh/2023/02/23/copyright-office-no-copyright-protection-for-certain-ai-generated-works/`.

[506] Vanessa Volz, G¨unter Rudolph, and Boris Naujoks. Demonstrating the feasibility of automatic game balancing. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, pages 269–276, 2016.

[507] Vanessa Volz, Boris Naujoks, Pascal Kerschke, and Tea Tušar. Tools for landscape

analysis of optimisation problems in procedural content generation for games. *Applied Soft Computing*, 136:110121, 2023.

[508] Kurt Vonnegut. Kurt vonnegut lecture. URL `https://www.youtube.com/watch?v=4_RUgnC1lm8`. YouTube video.

[509] Günter Wallner. Play-graph: A methodology and visualization approach for the analysis of gameplay data. In *8th International conference on the Foundations of digital games (FDG2013)*, pages 253–260. Foundations of Digital Games, 2013.

[510] Wei Wang and Ran Zhang. Improved game units balancing in game design through combinatorial optimization. In *2021 IEEE International Conference on e-Business Engineering (ICEBE)*, pages 64–69. IEEE, 2021.

[511] Yi Wang, Qian Zhou, and David Ledo. Storyverse: Towards co-authoring dynamic plot with llm-based character simulation via narrative planning. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, pages 1–4, 2024.

[512] Toh Weimin. A multimodal discourse analysis of video games: A ludonarrative model. *Proceedings of DiGRA 2015: Diversity of Play: Games–Cultures–Identities*, 2015.

[513] Joseph Weizenbaum. Computer power and human reason: From judgment to calculation. 1976.

[514] Yunge Wen, Chenliang Huang, Hangyu Zhou, Zhuo Zeng, Chun Ming Louis Po, Julian Togelius, Timothy Merino, and Sam Earle. All stories are one

story: Emotional arc guided procedural game level generation. *arXiv preprint arXiv:2508.02132*, 2025.

[515] Wim Westera, Rui Prada, Samuel Mascarenhas, Pedro A Santos, João Dias, Manuel Guimarães, Konstantinos Georgiadis, Enkhbold Nyamsuren, Kiavash Bahreini, Zerrin Yumak, et al. Artificial intelligence moving serious gaming: Presenting reusable game ai components. *Education and Information Technologies*, 25:351–380, 2020.

[516] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*, 2023.

[517] Jennifer R Whitson. What can we learn from studio studies ethnographies?: A "messy" account of game development materiality, learning, and expertise. *Games and Culture*, 15(3):266–288, 2020.

[518] Christopher D Wickens and Yili Liu. *Designing for people: an introduction to human factors engineering*. Createspace Independent Publishing Platform, 2017.

[519] T. Wijman. The games market and beyond in 2021: The year in numbers, 2021. URL `https://newzoo.com/insights/articles/the-games-market-in-2021-the-year-in-numbers-esports-cloud-gaming`.

[520] Paul Williamson and Christopher Tubb. Modelling player combat behaviour for dynamic difficulty scaling and combat perception analysis in first person shooter

games. *International Journal on Advances in Systems and Measurements*, 16(1–2):66–79, 2023. URL https://www.iariajournals.org/systems_and_measurements/sysmea_v16_n12_2023_paged.pdf.

[521] Brian M Winn. The design, play, and experience framework. In Richard E. Ferdig, editor, *Handbook of Research on Effective Electronic Gaming in Education*, pages 1010–1024. IGI Global, 2009.

[522] Winning Moves. Top trumps. [PAPER], 1978.

[523] Terry Winograd, Fernando Flores, et al. *Understanding computers and cognition: A new foundation for design*, volume 335. Ablex publishing corporation Norwood, NJ, 1986.

[524] Oliver Withington and Laurissa Tokarchuk. The right variety: Improving expressive range analysis with metric selection methods. In *Proceedings of the 18th International Conference on the Foundations of Digital Games*, pages 1–11, 2023.

[525] Oliver Withington, Michael Cook, and Laurissa Tokarchuk. On the Evaluation of Procedural Level Generation Systems. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, pages 1–10, Worcester MA USA, May 2024. ACM. ISBN 9798400709555. doi: 10.1145/3649921.3650016. URL https://dl.acm.org/doi/10.1145/3649921.3650016.

[526] Wizet. Maplestory. [DIGITAL], 2005.

[527] Weiqi Wu, Hongqiu Wu, Lai Jiang, Xingyuan Liu, Jiale Hong, Hai Zhao, and

Min Zhang. From role-play to drama-interaction: An llm solution. *arXiv preprint arXiv:2405.14231*, 2024.

[528] Peter Xenopoulos, João Rulff, and Claudio Silva. Ggviz: Accelerating large-scale esports game analysis. *Proceedings of the ACM on Human-Computer Interaction*, 6(CHI PLAY):1–22, 2022.

[529] Junjie H Xu, Zhou Fang, Qihang Chen, Satoru Ohno, and Pujana Paliyawan. Fighting game commentator with pitch and loudness adjustment utilizing highlight cues. In *2021 IEEE 10th Global Conference on Consumer Electronics (GCCE)*, pages 366–370. IEEE, 2021.

[530] Su Xue, Meng Wu, John Kolen, Navid Aghdaie, and Kazi A Zaman. Dynamic difficulty adjustment for maximized engagement in digital games. In *Proceedings of the 26th international conference on world wide web companion*, pages 465–471, 2017.

[531] Wataru Yaguchi and Makoto Murakami. Promotion of enjoyment by balancing competitive play in racing videogame. In *Proceedings of the 2023 International Conference on Power, Communication, Computing and Networking Technologies*, pages 1–5, 2023.

[532] Khurram Yamin, Shantanu Gupta, Gaurav R. Ghosal, Zachary C. Lipton, and Bryan Wilder. Failure modes of llms for causal reasoning on narratives, 2025. URL `https://arxiv.org/abs/2410.23884`.

[533] Daijin Yang, Yanpeng Zhou, Zhiyuan Zhang, Toby Jia-Jun Li, and Ray LC. Ai as

an active writer: Interaction strategies with generated text in human-ai collaborative fiction writing. In *Joint Proceedings of the ACM IUI Workshops*, volume 10, pages 1–11. CEUR-WS Team, 2022.

[534] Leni Yang, Xian Xu, XingYu Lan, Ziyan Liu, Shunan Guo, Yang Shi, Huamin Qu, and Nan Cao. A design space for applying the freytag's pyramid structure to data stories. *IEEE Transactions on Visualization and Computer Graphics*, 28(1): 922–932, 2021.

[535] Georgios N Yannakakis and Julian Togelius. Experience-driven procedural content generation. *IEEE Transactions on Affective Computing*, 2(3):147–161, 2011.

[536] Georgios N Yannakakis and Julian Togelius. Player modeling. In *Artificial Intelligence and Games*, pages 315–335. Springer, 2025.

[537] Casey Yano. 'slay the spire': Success through market analysis. Game Developers Conference (GDC) Vault, 2019. URL `https://www.gdcvault.com/play/1025667/-Slay-the-Spire-Success`.

[538] Shahram Yazdani, Armin Shirvani, and Peigham Heidarpoor. A model for the taxonomy of research studies: A practical guide to knowledge production and knowledge management. *Archives of Pediatric Infectious Diseases*, In Press, 05 2021. doi: 10.5812/pedinfect.112456.

[539] Nick Yee. *The Proteus paradox: How online games and virtual worlds change us-and how they don't*. Yale University Press, 2014.

[540] Nick Yee and Jeremy Bailenson. The proteus effect: The effect of transformed

self-representation on behavior. *Human communication research*, 33(3):271–290, 2007.

[541] Chris Young. Msn, January 2024. URL `https://www.msn.com/en-us/news/technology/nvidias-ai-npcs-enable-endless-life-like-conversations-in-video-games/ar-AA1mSuNV`.

[542] Bahram Hooshyar Yousefi and Hana Mirkhezri. Toward a game-based learning platform: a comparative conceptual framework for serious games. In *2019 International Serious Games Symposium (ISGS)*, pages 74–80. IEEE, 2019.

[543] Kristen Yu and Nathan R. Sturtevant. Application of retrograde analysis on fighting games. In *2019 IEEE Conference on Games (CoG)*, pages 1–8, 2019. doi: 10.1109/CIG.2019.8848062.

[544] José P Zagal and Roger Altizer. Examining'rpg elements': Systems of character progression. In *FDG*, 2014.

[545] ZA/UM. Disco elysium. [Windows], 2019.

[546] Mohammad Zohaib. Dynamic difficulty adjustment (dda) in computer games: A review. *Advances in Human-Computer Interaction*, 2018(1):5681652, 2018.

[547] Alexander Zook and Mark Riedl. A temporal data-driven player model for dynamic difficulty adjustment. In *Proceedings of the AAAI conference on artificial intelligence and interactive digital entertainment*, volume 8, pages 93–98, 2012.

[548] Alexander Zook, Eric Fruchter, and Mark O. Riedl. Automatic playtesting for

game parameter tuning via active learning. *CoRR*, abs/1908.01417, 2019. URL http://arxiv.org/abs/1908.01417.

[549] Robert Zubek. *Elements of game design*. MIT Press, 2020.

...Wait, what are you still doing here? There's regrettably no post-credits stinger for a sequel. Go play a game, have a snack, or get some rest. Balance in life is important, too.