The Case for Dynamic Difficulty Adjustment in Games

Robin Hunicke
Northwestern University
1890 Maple Avenue
Evanston, IL 60201
+01 847.491.7535
hunicke@cs.northwestern.edu

ABSTRACT

Conventional wisdom suggests that while players enjoy unpredictability or novelty during gameplay experiences, they will feel "cheated" if games are adjusted during or across play sessions. In order for adjustment to be effective, it must be performed without disrupting or degrading the core player experience. This paper examines basic design requirements for effective dynamic difficulty adjustment (DDA) given this constraint, presents an interactive DDA system (Hamlet), and offers preliminary evaluation results which challenge common assumptions about player enjoyment and adjustment dynamics.

Categories and Subject Descriptors

I.2.1 [**Artificial Intelligence**]: Applications and Expert Systems – *Games.*

General Terms

Design, Experimentation, Human Factors.

Keywords

Computer Entertainment, Video Games, AI, Dynamic Difficulty Adjustment.

1. INTRODUCTION

Games are boring when they are too easy, but frustrating when they are too difficult. In-game simulations (physics, graphics, AI) are increasingly complex, reacting in real-time to player action. However, game difficulty is still relatively static, which can lead to mismatches between player skill and game challenges. Dynamic Difficulty Adjustment (DDA) offers an alternative—modulating in-game systems to respond to a particular player's abilities over the course of a game session.

1.1 Game Design

Video games are designed to provide compelling experiences; players create these experiences by interacting with the game's internal systems. One such system is inventory—the stock of items that a player collects and carries throughout a game world. The relative abundance or scarcity of inventory items has a direct impact on the player's experience. As such, games are explicitly designed to manipulate the exchange of resources between world and player [12]. This network of producer-consumer relationships can be viewed as an economy—or more broadly, as a dynamic system [3][9].

1.2 Game Adjustment

Game developers iteratively refine game systems based on play testing feedback—tweaking them until the game is balanced. While this process can't be automated, directed mathematical analysis can reveal deeper structures and relationships within a game system. With the right algorithms, it is possible to adjust everything from a game's narrative structure [10], to the physical layout of maps or levels [16], while the game is being played.

In this work, we examine how DDA can be used to regulate a game's underlying economy. We explore how DDA can affect player progress by manipulating the supply and demand of various items. In addition, we discuss how these adjustments affect the player's perception of game difficulty, and their experience of frustration or enjoyment.

1.3 Design Constraints

Balance is critical to games [1]. Any system that can be taken advantage of or "gamed" by players will ruin a game's overall balance. This is most prominent in online games where the introduction of a new weapon or ability can quickly upset entire player communities.

In addition to balance, games must also provide consistent, reliable feedback. If internal systems have unintended consequences, they can inadvertently thwart player goals and aspirations. The "Rubber Band" adjustment AI featured in many commercial racing games is a common straw man for this very reason. When the pacing mechanism "cheats" by rocketing the pace car ahead in the final moments of a race, players are inspired to exploit it in order to win.

Successful DDA systems must maintain a game's internal balance and feedback mechanisms. They are at best unobtrusive, and at worst, inscrutable.

1.4 The Player Experience

To maintain balance and feedback, DDA systems must be built to accommodate a game's fundamental design goals with respect to player experience [4]. In order to adjust this experience for a given player (and avoid fixing what isn't broken), it is necessary to understand how it is designed (and why it is "fun"). The MDA framework (Mechanics, Dynamics and Aesthetics) helps us accomplish this goal [7].

Mechanics

Examining the basic player experience of a First Person Shooter (FPS) game, we can see it involves a continual process of trial and error—loops of exploring, fighting, triumph and death [Figure 1]. The connection between perception and action in an FPS is smooth and fast, leading to "twitch" or "fingertip" gameplay.

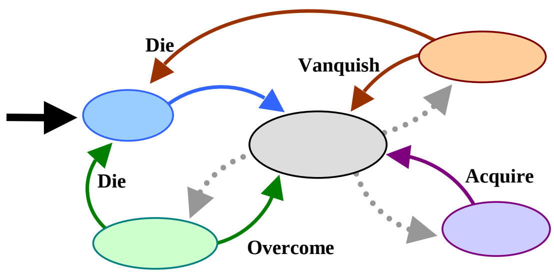

Figure 1. A simple state transition diagram for player activity in the First-Person Shooter genre of video games. Spawning is the start state, and dotted lines represent "finding" (obstacles, items or enemies, respectively).

Dynamics

As the player moves forward from encounter to encounter, and level to level, the difficulty and variability of obstacles increases. At the same time, rewards may become increasingly scarce, variable and valuable—so that the player must work harder to gain new (more powerful) weapons, health, power-ups, and so forth. This ramping challenge is the fundamental *dynamic* of the game.

Aesthetics

Overall, the mechanics and dynamics of FPS games work to create an active, fluctuating economy: a set of producer-consumer relationships which work to shape the player's experience of difficulty. Generally speaking, dedicated players in the FPS genre are trained (via previous exposure) to *expect* these economic dynamics. They understand that over time, aid will taper in frequency as challenge and variability increase. Together, these two dynamics create the overarching *aesthetics* of the FPS genre.

1.5 DDA with MDA

Using the MDA framework as a guide, we have created a system that helps reduce unnecessary negative feedback, without intruding upon the basic FPS gameplay experience [6]. We have designed systems that regulate the game's core inventory *Mechanics* (health, ammunition, shielding and weapons), which effect the game's primary exploration and combat *Dynamics*, while maintaining the overall cycle of action (which contributes to the game's fast-paced shooter *Aesthetics*).

This implementation does not alter the fundamental mechanics or aesthetic goals of the target game application. Players can die, because it is a critical part of the game's overall design. As such, the original game balance and feedback mechanisms are preserved [5].

2. EXAMPLE SYSTEM

The Hamlet system is comprised of functions integrated with the *Half-Life* SDK [17]. These functions monitor the exchange of inventory items over the course of a gameplay session, and modify the game's behavior according to some overarching adjustment goal.

In the most general sense, the goal of Hamlet's DDA algorithm is to map the current state of the game world to some set of adjustment actions. This is accomplished by an evaluation function (which maps from the state of the game world to an evaluation of player's performance), and an adjustment policy, (which maps this evaluation to some set of adjustments in the game world).

At first glance, these functions can be quite complex. However, by focusing on specific producer-consumer relationships (and in particular, the production and consumption of player health), we can simplify this calculation as follows.

2.1 Estimating Probability of Death

We begin by establishing the probability distribution of damage done to the player by her enemies during combat. By taking a series of measurements over time, we can establish that this distribution assumes the form of a Gaussian.

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

The Gaussian error integral allows us to calculate $F(d)$, the probability of receiving d or less damage on a given tick.

$$F(d) = \int_{-\infty}^{d} p(x)dx = 1\Big/\left[\sigma\sqrt{2\pi}\right]\int_{-\infty}^{d} e^{(x-\mu)^2/(2\sigma^2)}dx$$

This integral is approximated in C and C++ math library by *erf* function. We can continually calculate *erf(x)* in terms of the player's current health h, mean and standard deviation of current opponent damage rates, at some time in the future, t.

$$F(d_t) = 1 - \tfrac{1}{2}\left\{1 + erf\left(\frac{h-\mu t}{\sigma\sqrt{2t}}\right)\right\}$$

During combat, Hamlet records the damage d and squared damage d^2 each enemy does to the player, by class.[1] The player's

[1] Taking damage measurements across all game time would artificially lower the average expected damage by including many

probability of death is then calculated using μ and • derived for each enemy present in the encounter.[2] By tracking just a few numbers, we can predict the likelihood of player death in a given encounter – which in turn helps us decide when to intervene.

2.2 Control Policy Design

Intervention is conducted by a control policy, which is designed to affect the underlying game economy based on estimated player performance. Each control policy consists of an estimation algorithm (in this case, the damage estimator outlined above), an adjustment goal (determined by the desired player experience), and a set of intervention strategies.

Adjustment Goals

Adjustment goals help parameterize a given policy based on the desired player experience (or aesthetic goal). For example, a simple *comfort* policy aims to keep players feeling challenged, but safe—padding their inventory so that they stay within some mean range of health over the course of a game.

A *discomfort* policy is optimized to challenge players by limiting supply and driving up demand whenever they reach some designated range of health (or when they stay at a given level for some period of time). A t*raining* policy is somewhere between the two – initially comforting the player, gradually adjusting the difficulty to increase discomfort over the course of a level or session.

This is by no means an exhaustive list of policies. And while we leave the specifics of policy construction to the discretion of the designer/researcher, it is important to note that the adjustment goal is a critical component of any DDA controller. Without the proper aesthetic considerations, adjustment runs the risk of thwarting player expectations, competing with existing experience goals, and disrupting the player's immersion or suspension of disbelief.

Intervention Strategies

Intervention strategies manipulate the game mechanics to affect the difficulty dynamics in order to achieve the adjustment goal. In this work, we focus on the mechanics governing supply and demand.

On the supply side, interventions can manipulate the player's inventory by placing items in the playfield (health packs, ammunition, and weapons). Other interventions include modifying the player's hit points, the strength of their attacks and weapons, their accuracy, and properties of items in the playfield or player inventory.

On the demand side, interventions can manipulate the impact of enemies by changing their class, number or hit points, weapon or location (before spawn). They may also adjust enemy hit points, attack strength, accuracy or AI states and so on.

2.3 Implemented Policy

For the purposes of evaluation, we implemented a simple "comfort zone" policy , which works to keep players a relatively safe distance from death. During combat, if the estimator determines that the player's probability of death is greater than 40%, it begins to intervene.

The goal of the example policy is to keep the player's mean health at 60, with a standard deviation of 15 points. During combat, the policy will add health in 15 point segments, at 100 tick intervals.

Hamlet is designed to actively help struggling players, without "propping them up". Threshold values for adjustment, intervention increment and lag were all set based on user testing and observation with respect to this goal. In addition, we wanted to disguise the adjustment as much as possible while earnestly reporting environmental changes to the player's HUD. Speed worked to our advantage here (see discussion below in 3.2).

3. EXPERIMENTAL EVALUATION

3.1 Goals

The aim of this study was to show that dynamic difficulty adjustment can be performed effectively, without disrupting or degrading the core gameplay experience. We designed the study to answer these basic questions:

- Does adjustment affect player performance?
- Do players notice adjustment?
- Does adjustment significantly affect the player's enjoyment, frustration or perception of game difficulty?

3.2 Setup

The experiment was performed using a Dell Dimension 8200 with a 1.8 gigahertz Intel Pentium processor, 1 gigabyte of RAM and an NVidia GeForce3 graphics card. The test game was *Case Closed,* a custom game built for the *Half-Life* engine [18]. We made no cosmetic changes to the inventory feedback system. Fluctuations in health and ammo were reported precisely as they would be in any game.

During play, probability calculations are performed on every tick whenever the user is in combat, running in real time with no visible impact on the game's overall performance or frame rate. The main code loop takes 125,000 CPU cycles – running at an average of 69 microseconds per tick. Of that time – approximately 0.6 microseconds are required for the *erf* calculation.

3.3 Experimental Protocol

Subjects were solicited via flyers within the Computer Science building and on several undergraduate mailing lists. Before

---

time samples where there was no activity—including player exploration, backtracking, and pauses. As such, we mark combat with our first detection of damage (to player or enemy) or the movement of an enemy into an active state (as when the player trips a trigger volume).

[2] To account for early combat situations (where the player has not yet encountered a given class of enemy), all classes are initialized with a default d and d^2 (10, and 100 respectively). The look-ahead t is 300 ticks (about 10 seconds). These initial values are based on play testing and observation, but could also be supplied by a designer, or generated via more complex machine learning techniques [13].

proceeding with the experiment, each candidate read and signed a consent form which stated plainly that their play data was being recorded for evaluation purposes. Details of the system and evaluation goals were not disclosed at this time. However, the experiment title included the phrase "Dynamic Game Adjustment".

After reading a game overview (which explained controls and back story), subjects were seated at a PC where they played the game for at least 15 minutes. Performance data was logged in text files and stored for later evaluation. After completing the experiment, subjects filled out an evaluation form (all ratings based on a 5 point scale, with 1 and 5 labeled low and high respectively), signed up for a optional raffle, and were debriefed about their play experience. The experiment was single-blind, with half of the games adjusted and the other half control.

4. RESULTS

Table 1: Preliminary results given an exploratory sample of 20 subjects of mixed ability (novice to expert). Results presented have a t-value of 2.09246 and P of 0.050836.

	Mean	Std. Deviation
Unadjusted	6.400000	2.108185
Adjusted	4.000000	2.951459

4.1 Adjustment

Preliminary findings show a mean value of 6.4 player deaths in the first 15 minutes of unadjusted games, with a standard deviation of 2.1. In adjusted games, players died an average of 4.0 times in the first 15 minutes of play, with a standard deviation of 2.9. Post-play evaluations of adjusted games reveal no significant correlation between adjustment and enjoyment for novice players. However, trends indicate that expert players report slightly elevated levels of enjoyment.

4.2 Perception of Adjustment

Overall, there was only a small correlation (0.3) between the perception and actuality of game adjustment. Subjects often perceived adjustments that were not there, and several discounted adjustment when in fact they had been receiving help throughout the game.

Interestingly enough, most subjects did have suggestions for what the game might do to help them. These included better lighting and sound cues, help with weapons and targeting, hints about the location of objects, or behavior of obstacles and enemies. Of all the subjects tested, only two remarked that adjusting a game's difficulty would lessen their enjoyment. Of these two trials, one was adjusted and the other was not—but neither reported it,

4.3 Self Perception

There was little correlation between a player's self-rating (with respect to games in general or *Case Closed* in particular) and their actual performance. In both adjusted and unadjusted games, several players who rated themselves relatively high (3-5) performed on par with others who had rated themselves as novices (1-2), and vice-versa.

Regardless of aptitude or exposure, it seems, people are likely to assume that they have failed to meet the standard for expert (5) or even good (3) performance. In the post-test debriefing, several subjects bemoaned their poor performance, despite ranking in the top third.

4.4 Difficulty Perception

Subjects' perceptions of difficulty did not correlate with their self-rating, either. Experts familiar with Half-Life and novices who rarely play shooters both rated the game as somewhat to extremely difficult (3-5). As of yet, there is no significant correlation between adjustment and difficulty evaluation.

5. ANALYSIS

5.1 Change Blindness

In the last decade, vision researchers have been exploring the phenomenon of change blindness – whereby observers of a scene fail to detect obvious changes when they occur during saccades (eye movement), "wipes" or "mudsplashes" [11].

Figure 2: Change blindness techniques will mask the transition of objects such that users have a very difficult time noticing the change from a solid to dashed line in the image above.[8]

Change blindness occurs for simple changes (such as objects appearing or disappearing) as well as radical changes (actors swapping heads between scenes in a video, virtual blocks changing height while a subject is manipulating and sorting them) [14]. It is not surprising, then, that subjects did not perceive changes to their health during combat, as their attention was directed elsewhere.

It has already been suggested that certain interactive applications take advantage of change blindness—for example, to reduce distracting visual changes in ubiquitous information displays [8]. Given the indications of this study, it seems equally valid to suggest the use of change-blind strategies for game adjustment. If nothing else, directed research can tell us how accurately players track objects within game environments and how closely they correlate gameplay or system behavior with visual cues.

5.2 Frustration, Enjoyment and Adjustment

It would seem that novice players assume a low success rate regardless of actual performance, while experts are likely to attribute any and all success to their own prowess. In fact, novice users consistently rated themselves as having to repeat tasks (5) regardless of success—perhaps because they were not familiar with the repetitive nature of FPS games.

For this and other reasons, it is not clear whether the adjustment techniques explored here can impact the enjoyment of players unfamiliar with the FPS genre. However, if something this basic can increase a familiar player's enjoyment without ruining their sense of agency or control, imagine what a professionally designed system might contribute to their gameplay experience!

6. FUTURE WORK

At its core, adjustment is about economics: manipulating the supply and demand of items in a given play scenario. As such, we believe that the techniques used in Hamlet will scale beyond simple FPS games to any system with a clear economy of items—including RPG and strategy games as well as simpler arcade-style interactions. We are currently extending the DDA techniques expressed in this paper for use in Neverwinter Nights [19].

But there are other, emerging applications for DDA. In his recent address at the Game Developers Conference 2005, Will Wright stated that the rising cost of content production is of grave concern for game developers. Dynamic content, he maintained, will allow users to experience a myriad of game elements while freeing developers from the strain of creating static levels and obstacles [15].

How can the methods explored here be used to balance such complex content systems? Whether scheduling content downloads, organizing encounters or managing the player's overall progress, dynamic adjustment algorithms can help clarify and simplify this new form of game production. And by breaking previously ad-hoc design decisions down into their component mechanical implications, it *is* possible to build game systems that provide novel, well-balanced, dynamic gameplay.

7. CONCLUSION

Traditionally, discussions of DDA and dynamic game adjustment are met with resistance. Conventional wisdom about adjustment has led many to question the concept and shy away from the up-front costs of developing sound adjustment algorithms.

This study indicates that even crude adjustment algorithms can improve performance, while retaining the player's sense of agency and accomplishment. Furthermore, when designed to work with a game's fundamental MDA, adjustment of even the most basic gameplay elements is nearly imperceptible to the player. Change blindness facilitates this imperceptibility.

Is dynamic adjustment worth the effort for commercial developers? Will DDA make games more enjoyable for "average" or "novice" players? It is time to test our assumptions about players and play. With rigorous study and evaluation, we can understand the fundamental limits—and true potential—of these techniques.

8. REFERENCES

[1] Adams, E., Balancing Games with Positive Feedback. Gamasutra.com, 2002.

[2] Bolton, J., Using Lanchester Attrition Models to Predict the Results of Combat in K. Pallister, ed., *Game Programming Gems 5*, Massachusetts: Charles River, 2005.

[3] Castronova, E., Virtual Worlds: A First-Hand Account of Market and Society on the Cyberien Frontier. *Working Paper, Center for Economic Studies and IFO Institute for Economic Research*: Munich, Germany, 2001.

[4] Church, D., Formal Abstract Design Tools. *Game Developer*, August 1999. San Francisco, CA: CMP Media.

[5] Hunicke, R., *Designing Dynamic Difficulty Adjustment for Games.* Ph.D. Thesis, Northwestern University, Evanston, IL, (forthcoming 2005).

[6] Hunicke, R., Chapman, V., AI for Dynamic Difficulty Adjustment in Games. In *Proceedings of the Challenges in Game AI Workshop, Nineteenth National Conference on Artificial Intelligence (AAAI '04)* (San Jose, California) AAAI Press, 2004.

[7] Hunicke, R., LeBlanc, M, and Zubek, R., MDA, A Formal Approach to Game Design and Game Research. In *Proceedings of the Challenges in Game AI Workshop, Nineteenth National Conference on Artificial Intelligence (AAAI '04)* (San Jose, California) AAAI Press, 2004.

[8] Intille, S. Sl, Change Blind information Display for Ubiquitous computing Environments. *Proceedings of the 4th International Conference on Ubiquitous Computing*, (Göteborg, Sweden) Springer-Verlag, 91-106, 2002

[9] Luenberger, D. G., *Introduction to Dynamic Systems: Theory, Models, and Applications*. New York: John Wiley and Sons, Inc, 1979.

[10] Mateas, M.. *Interactive Drama, Art, and Artificial Intelligence*. Ph.D. Thesis. Technical Report CMU-CS-02-206, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. December 2002.

[11] Simons, D. J., Current Approaches to Change Blindness. *Visual Cognition, 2000, 7* (1/2/3), 1-15.

[12] Simpson, Z., The In-game Economics of Ultima Online, *presentation at Game Developers Conference, San Jose, March 9-14, 2000.*

[13] Spronck,P., I. Sprinkhuizen-Kuyper and E. Postma, Difficulty Scaling of Game AI. *GAME-ON 2004: 5th International Conference on Intelligent Games and Simulation* (eds. El Rhalibi, A., and D. Van Welden), pp. 33-37. EUROSIS, Belgium, 2004.

[14] Triesch, J., Ballard, D. H, Hayhoe, M. M., Sullivan, B. S., What you see is what you need. *Journal of Vision, 2003, 3,* 86-94.

[15] Wright, W., The Future of Content, *presentation at the Game Developers Conference, San Francisco, March 7-12, 2005.*

[16] Zehnder, S. *Wayfinding, Maps, and Visual Search in Video Games*. [Dissertation Manuscript], Northwestern University. (Forthcoming, 2006).

[17] *Half-Life*, Valve Software, 1998.

[18] *Case Closed*, Delrue, S. et al, 2001.

[19] *Neverwinter Nights*, Bioware, 2002.