DYNAMIC DIFFICULTY ADJUSTMENT IN UE5 BOSS FIGHTS: INTEGRATING PLAYER VELOCITY TRACKING AND HYBRID FEEDBACK SYSTEM

INTERNATIONAL JOURNAL OF ARTIFICIAL INTELLIGENCE AND SUSTAINABLE TECHNOLOGIES
VOL 1 (01), JULY-DECEMBER 2026

Gaurav Savita
Department of IT JSS university Noida, India
Email: j240212040@jssuninoida.edu.in
Krish Gupta
Department of CSDS JSS university Noida, India
Email: j240213024@jssuninoida.edu.in
Jaspreet Kaur
Department of CSE JSS university Noida, India
Email: jaspreetkaur@jssuninoida.edu.in

ABSTRACT

This paper proposes a novel framework for Dynamic Difficulty Adjustment (DDA) specifically tailored for climactic boss encounters in Unreal Engine 5 (UE5). Moving beyond traditional metrics, the framework introduces player velocity as a primary implicit performance indicator, leveraging its potential to signal nuanced player states such as aggression, panic, or fatigue. The system integrates velocity data within a hybrid feedback loop that combines real-time game telemetry (implicit feedback) with post-encounter player ratings (explicit feedback). A weighted heuristic algorithm is proposed to process these diverse data streams and dynamically adjust the boss AI behavior, attributes, and environmental factors to maintain a continuous state of flow for the player. This paper details the technical implementation within UE5 using C++ and Blueprint, proposes a controlled experimental protocol for future validation, and discusses the ethical considerations of DDA. The proposed system is designed to create a more organic, invisible, and personalized challenge that could mitigate the common player perception of 'scripting' and 'cheating.

Keywords: *Dynamic Difficulty Adjustment, DDA, Unreal Engine 5, UE5, Player Velocity, Hybrid Feedback, Game AI, Boss Fights*

1. INTRODUCTION

Background: The Problem with Static Difficulty

The design of a compelling and challenging boss fight is a cornerstone of modern video games, serving as a climactic, skill-gating encounter that tests the player's mastery of the game's mechanics. Historically, game difficulty has been managed through a static model in which players select a fixed challenge level, such as "Easy," "Medium," or "Hard," at the start of their journey. However, this approach fundamentally fails to account for the immense variability in individual player skills and learning curves. A challenge that is perceived as difficult for one player may be easy for another, and vice versa. This static model often leads to two detrimental player states that developers aim to avoid: When a boss fight's challenge is too high for a player's skill, it can result in anxiety and frustration, potentially becoming an insurmountable obstacle that leads to player disengagement or a "rage quit." Conversely, if the challenge is too low, the player's skills will outpace the difficulty, leading to a state of boredom and detachment. In either scenario, the game's pacing is undermined, and the player's investment and satisfaction are significantly reduced.

The Promise of Dynamic Difficulty Adjustment (DDA)

Dynamic Difficulty Adjustment (DDA), also referred to as adaptive difficulty or dynamic game balancing, is a game design technique that addresses the limitations of static difficulty by modifying game parameters in real time to match the player's performance and skill level. The core psychological objective of DDA is to maintain a player in a state of "Flow," a concept defined by psychologist Mihaly Csikszentmihalyi as a mental state of energetic focus in which a person is completely immersed in an activity. By dynamically adjusting the challenge to evolve with the player's abilities, DDA aims to keep them continuously engaged and within this optimal state of experience.

However, the implementation of the DDA is fraught with significant paradoxes. Although its intention is to enhance the player experience, a poorly implemented system can actively degrade it. This occurs when players perceive game adjustments as "scripting," "cheating," or overt manipulation of the game's rules to engineer a specific outcome. This perception shatters the illusion of a fair challenge, as if the game is not a consistent system to be mastered but a manipulative opponent to be outsmarted by the player. The problem is magnified when a player who is performing well suddenly feels that the game is fighting back unfairly, or when a player in a state of struggle is too obviously patronized with an influx of easy victories. Therefore, a successful DDA system must not only be effective at balancing difficulty but also be subtle, invisible, and inform consistent. Its changes must feel organic, as if the challenge itself is a living entity that is naturally reacting to the player's evolving strategy rather than a robotic algorithm

applying a linear correction.

Proposed Solution and Research Questions

This paper proposes a novel DDA framework for UE5 boss fights that directly addresses the DDA paradox by leveraging a subtle yet powerful metric: the player velocity. Velocity tracking, which is typically used in sports analytics to monitor an athlete's physical state, offers a unique window into a player's psychological state during the game. The proposed system integrates implicit real-time velocity data with explicit subjective player feedback collected via post-encounter surveys. This hybrid feedback model informs a weighted heuristic algorithm that dynamically adjusts the boss AI behavior and other gameplay parameters.

The central hypothesis is that an adaptive system using this hybrid model will lead to a more consistent, subtle, and enjoyable boss-fight experience compared to a static difficulty system. To explore this, the following research questions were posed:

- Can player velocity serve as a valid and reliable indicator of a player's psychological state and performance in a boss fight?
- How can a hybrid feedback system, blending implicit telemetry with explicit survey data, be mathematically modelled to inform a DDA algorithm?
- Does the proposed DDA framework, when implemented in UE5, demonstrably improve player satisfaction and engagement compared to a static difficulty encounter?

2. LITERATURE REVIEW: THE THEORETICAL FOUNDATIONS OF ADAPTIVE GAMEPLAY

The Psychology of Flow and DDA

The concept of flow, as articulated by psychologist Mihaly Csikszentmihalyi, is a cornerstone of modern game design and the foundational principle behind DDA. Flow represents a state of deep, effortless engagement that occurs when a person's skill level is perfectly matched to the challenge of an activity. Games that fail to achieve this equilibrium are often categorized into two extremes: boredom, which arises when a player's skills surpass a game's trivial challenges, and anxiety or frustration, which occurs when challenges are too daunting for a player's abilities. DDA's primary goal of DDA is to guide the player into and maintain them within this "flow channel," continuously adjusting the game's challenge to match their evolving skill set.

A common misconception in DDA design is that the system should always aim for balanced, moderate difficulty to achieve flow. However, some research suggests a more nuanced relationship between difficulty and enjoyment in games. One study found that, despite a lower level of challenge, players enjoyed the easiest game version the most, which allowed for a higher number of "success experiences" (e.g., enemies killed) and a lower number of failures. This finding complicates the direct application of flow theory, as it suggests that players' subjective enjoyment may be more strongly tied to a feeling of mastery and accomplishment than to a perfectly balanced challenge. Therefore, a successful DDA system must not only prevent frustration but also ensure that the player feels a consistent sense of power and victory, which may sometimes involve making the experience slightly easier than a purely balanced statistical model would dictate [2].

A Taxonomy of DDA Methods

DDA systems employ various techniques to adapt gameplay, ranging from simple parameter manipulation to

complex AI-driven models. These methods can be categorized as follows.

Parameter Manipulation

This is the most straightforward DDA approach. This involves altering specific and quantifiable game elements. Common adjustments include changing the speed, health, or damage of enemies, modifying the frequency of powerups and resources, or altering a player's power and abilities [1]. For instance, the DDA system in Resident Evil 4 dynamically increases enemy aggressiveness and numbers when a player is performing well; conversely, it reduces enemy toughness or frequency when the player is struggling.as shown in table 1.

AI-Driven Methods

These are more sophisticated systems that dynamically alter the behaviour of non-player characters (NPCs) or an opponent's AI. Examples include a Monte Carlo Tree Search (MCTS) AI that progressively chooses less optimal actions to reduce its performance and make the game easier for the player. Reinforcement Learning (RL) is also used, where an AI's "exploration vs. exploitation" rate can be adjusted to influence its performance and maintain a player's flow. Procedural Content Generation (PCG): This method involves procedurally generating or modifying the level design to suit the player's skill. This could include adjusting the distance between platforms in a side scrolling game or varying the complexity of the race track [6].

Table 1: DDA Methods and Their Metrics

DDA Method	Core Mechanism	Typical Metrics Used	Examples of Application
Parameter Manipulation	Directly changes game variables like enemy stats or resource frequency	Damage dealt/taken, shots/hits, time to complete tasks, deaths, resource counts	Resident Evil
AI Driven Methods	Alters opponent behavior or strategy via algorithms	Win/loss rates, AI decision paths, player skill level (measured by events)	MCTS systems adjusting win probability, RL systems balancing exploration/ exploitation.
Procedural Content Generation (PCG)	Generates or modifies game levels and environments	Level completion time, number of deaths in an area, pathing efficiency	Adjusting platform distances, varying track complexity

The Role of Player Metrics in DDA

DDA systems are fundamentally reliant on continuously monitoring player performance using various metrics. Traditional metrics are often based on discrete events and outcomes. Examples include combat success rates, damage taken, deaths, and the time taken to complete tasks. This data, collected as game telemetry, provides an objective record of a player's actions and their consequences [5]. This study introduces player velocity as a valuable and novel metric for DDA. In sports analytics, velocity-based training (VBT) uses an athlete's movement speed to assess their physical state, such as fatigue or readiness for a workout. Similarly, in game design, player velocity offers a continuous and implicit indicator of a player's psychological state. During a chaotic boss fight, a player moving at a high peak-to-average velocity ratio, characterized by short, erratic bursts of speed, may exhibit signs of panic or anxiety. Conversely, a player with a high and consistent average velocity might confidently strafe and kite the boss, indicating a sense of mastery and control. A sudden, unprovoked drop in the average velocity could signify player indecision or confusion. Player velocity is not merely a measure of movement; it serves as a subtle proxy for the player's mental state, providing a continuous stream of information that complements the event-based data from traditional combat metrics. This allows a DDA system to detect when a player is slipping out of the flow channel, even before a tangible failure event, such as death or a missed attack, occurs [3].

The Spectrum of Feedback: Implicit vs. Explicit Systems

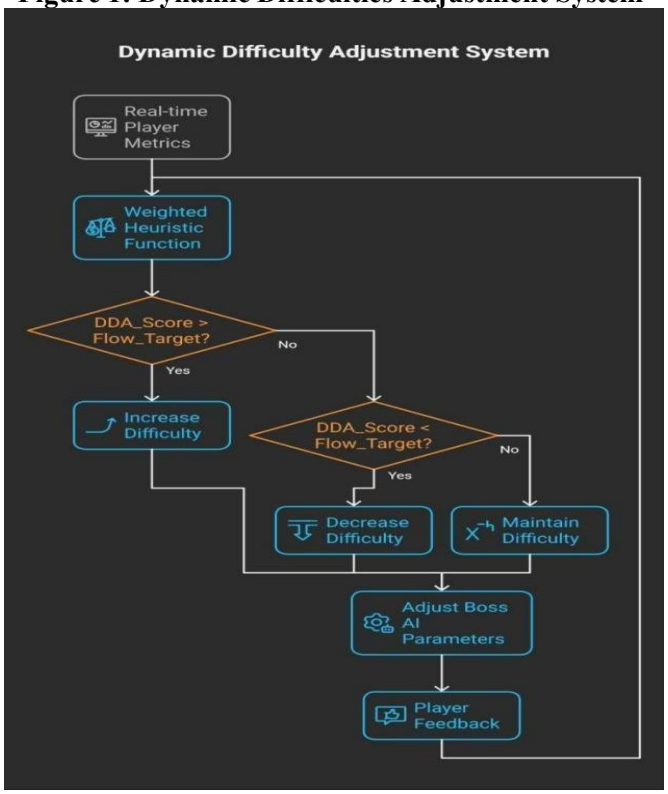
Feedback mechanisms in DDA systems operate on a spectrum, from objective real-time telemetry to subjective post-game surveys. Implicit feedback, or telemetry, involves realtime data collection in the background, such as the metrics discussed above. This data is objective and provides a continuous record of a player's actions and performance. Explicit feedback, on the other hand, is data directly provided by the player, typically through in game ratings, surveys, or prompts. This feedback is subjective and measures a player's perceived experience, satisfaction, and enjoyment [4]. A hybrid system combines both implicit and explicit feedback to create a more robust and accurate DDA model. A system based on purely implicit data can be fooled; for example, a player's objective performance might be high, yet their subjective feedback could reveal a lack of enjoyment owing to a perceived lack of challenge. A hybrid approach addresses this by using explicit subjective data to validate and calibrate implicit objective data. For instance, if a player who moved erratically and died frequently (poor implicit performance) consistently provided a high "enjoyment" rating on a post-encounter survey, the system learned that this playstyle was not necessarily a negative indicator. In subsequent encounters with similar players, the system can dynamically adjust its internal weighting function to place a lower value on negative implicit metrics, such as "deaths" and "velocity fluctuation," and a higher value on positive ones, such as "time spent in combat," focusing on keeping the fight engaging rather than simply preventing death. This creates a sophisticated, self-tuning algorithm that learns not only from what happened but also from how the player felt about it, moving beyond a purely reactive model to a predictive, player centric one [6].

3. A Framework for Hybrid Feedback DDA in UE5 Boss Fights

Conceptual Model: The Player-Boss Feedback Loop

The proposed DDA system operates in a continuous feedback loop. The player's actions are the primary input, which is then fed into a data collection subsystem that gathers various metrics in real time. These data are processed by a DDA Manager that houses a weighted heuristic algorithm. The algorithm calculates a Difficulty Score and determines an Adjustment Factor that is sent to the Boss AI Manager. The boss's behavior, attributes, and other parameters are modified, and its subsequent actions create a dynamic challenge that elicits a reaction from the player, thus completing the loop.as shown .

Figure 1: Dynamic Difficulties Adjustment System



The Player Velocity Sub-system: Data Acquisition and Interpretation

The acquisition of player velocity data in UE5 was straightforward. The Character Movement Component, a standard part of the engine's character classes, provides direct access to the velocity vector. This data can be easily accessed in both Blueprint and C++ by calling Get Velocity () on the movement component. For a robust, multiplayer enabled system, this data must be tracked on the server to prevent manipulation by malicious users [7]. The system tracks and analyzes three core metrics derived from the player velocity: [8,9].

Average Velocity: The mean speed of the player throughout the encounter. A consistently high average velocity may indicate an aggressive and confident playstyle, whereas a low average velocity could suggest a more cautious or uncertain approach.

Peak Velocity: The highest speed achieved in short bursts, indicative of high-intensity manoeuvres such as sprinting, dodging, or repositioning. The frequency and context of these bursts provide insights into how the player reacts to the boss's attack patterns.

Velocity Fluctuation: The standard deviation of the velocity over a time window. High fluctuation could signal erratic movement driven by panic or a lack of understanding of the boss's mechanics [8,9], whereas a low standard deviation suggests smooth, controlled traversal of the arena.[7]

Table 2

Met ric Cate gory	Metric	Descripti on	Data Type	Source
Imp licit	Averag e-Velocit y	Mean speed of the player throughou t the fight	Continuo us	Character Movement Component
	Peak Velocit y	The maximum	Continuo us	Character Movement Component
	Velocit y-Fluctua nt	Stand-deviation of	Continuo us	Character Movement Component
	Damag e Taken per Second	Total damage received by the player.	Continuo us	Game play Ability System
	Damag e Dealt per Second	Total damae inflictd on the boss.	Continuo us	Game playability System
	Time to Kill (TTK)	Time from the start of the fight until the boss is defeated.	Continuo us	Game State/Logger
	Numbe r ofDod ges/Spr ints	Count of player-initiated	Count	Event Listener
		high-intensity maneuver s		
	Resour ce Pickup s	Count of health, ammo, or power- ups collected.	Count	Event Listener
Expl icit	Perceiv ed Difficu lty	Player rating of how difficult the fight felt.	Scale (e.g., 1-10)	Post-encounter survey

Figure 2: Formula for velocity fluctuation

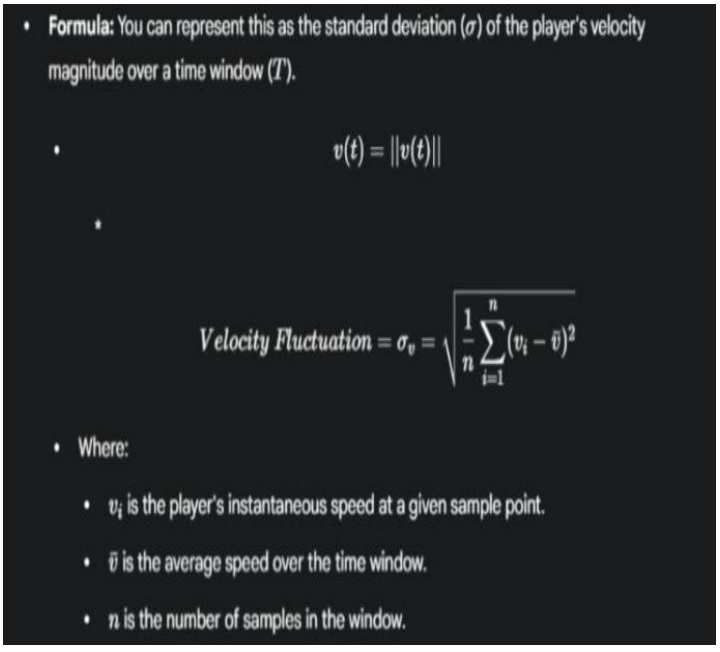


Table 3: Proposed Hybrid Feedback Metrics

	Perceiv ed-Enjoy ment	Player rating of how much they enjoyed	Scale (e.g., 1-10)	Post-encounter survey
	Perceiv Contro l	Player rating of how much-control they felt.	Scale (e.g., 1-10)	Post-encounter survey

The Hybrid Feedback System

The DDA system relies on a comprehensive set of metrics to build a complete profile of the player's performance and perceived experience. These metrics are categorized into implicit (telemetry) and explicit (survey) data types as shown in table 2. DDA Manager and its decision-making logic. The Game State, which is replicated to all clients, stores and syncs game-wide data, such as the current DDA difficulty level. Player-specific metrics, such as velocity, damage, and death count, are best managed within the Player State class, which is designed for per-player data and is replicated for all clients for real-time multiplayer updates. Continuous data collection and analysis, particularly of velocity, are computationally intensive and must be handled in C++ to avoid the performance overhead of blueprints and ensure predictable behaviour[13]

The DDA Algorithm: A Weighted Heuristic Function

The core of the DDA system is a weighted heuristic function that processes the collected metrics to produce a single DDA Score. This score represents the player's current performance relative to the desired challenge level. The algorithm is defined by the following formula: Where Mi represents each of the implicit metrics (e.g., average velocity and damage taken), and wi is a dynamic weighting factor assigned to each metric. Explicit feedback from post encounter surveys was used to dynamically adjust these weights. This is a crucial humanin-the-loop mechanism that prevents the DDA algorithm from optimizing purely statistical outcomes at the expense of player satisfaction [10-12]. For example, if a player who died frequently and moved erratically (metrics of poor performance) provided a high "enjoyment" rating, the algorithm learned that this playstyle was not necessarily a negative indicator. In subsequent encounters with similar players, the system might reduce the weight of "deaths" and "velocity fluctuation" and increase the weight of "time spent in combat," focusing on keeping the fight engaging rather than simply reducing the difficulty to prevent death. This creates a sophisticated, self-tuning algorithm that learns not only from what happened but also from how the player felt about it, moving beyond a simple reactive model to a predictive, player-centric one. The DDA_Score is then compared against a Flow_Target_Value[14] (e.g., a balanced win rate of approximately 50%), and the resulting Adjustment Factor is sent to the boss AI manager [5].

UNREAL ENGINE 5 IMPLEMENTATION AND TECHNICAL CONSIDERATIONS

A Data Collection and Telemetry

A robust implementation of this framework in UE5 requires a hybrid C++/Blueprint approach. While Blueprints are ideal for rapid prototyping and eventdriven logic, the continuous, real-time nature of DDA, especially in a networked environment, demand the performance and stability that a C++ foundation provides**.** The system architecture relies on the UE5 Gameplay Framework. Game Mode is a serveronly class that defines the core rules of the game and is an ideal place to manage the

Boss AI and Behaviour Trees

The boss's behaviour is driven by a Behaviour Tree, a powerful and flexible AI tool in UE5. The Behaviour Tree includes conditional decorator nodes that read a difficulty parameter managed by the DDA Manager. This parameter serves as a dynamic input for AI decision-making. For example, at a lower difficulty level, the boss might follow a predictable attack pattern with large "tells" (warning animations), but at a higher difficulty, it could introduce a new combo attack, increase its movement speed, or overlap different attacks to create a more challenging experience [15].

The DDA Manager Class

The DDA Manager is a central Actor Component that serves as the brain of the DDA system. It will encapsulate the core DDA logic and operate on a timed loop (e.g., every 5-10 seconds) to reduce processing cost and prevent “over-adjusting” the difficulty. Its main functions are as follows:

Collect Metrics(): This function gathers data from the Player State and other game systems, including the player's velocity, damage dealt, and damage taken.

Calculate Difficulty(): This function runs the weighted heuristic algorithm to compute the DDA_Score based on the latest metrics.

Apply Adjustment(): This function communicates the calculated Adjustment Factor to the boss AI, modifying its

Behaviour Tree variables, attack attributes, and other relevant parameters [15].

Practical Application

The implementation of the DDA system leverages a hybrid approach to achieve this. The core logic,

including the heuristic algorithm and metric collection, was implemented in C++. Designers would then use blueprints to visually script the AI's behaviour and link it to the C++ variables[16]. A simple blueprint graph for collecting player speed is as follows

4. PROPOSED RESEARCH METHODOLOGY

Experimental Protocol: A Controlled Study

To validate the proposed DDA framework, a quantitative semi-experimental study was conducted using a pretest/post-test design as shown in table 3. Participants will be randomly assigned to one of two groups: an Experimental Group (EG) exposed to the DDA system, and a Control Group (CG) that plays a single, static "Hard" difficulty version of the same boss fight. A cohort of 50 participants with varying levels of gaming experience will be recruited to ensure that the DDA system is tested across a diverse skill range.

The experimental protocol followed these steps:

Baseline: Participants completed a pre-session survey to establish their initial gaming experience and expectations.

Gameplay: Each participant played the same boss fight. The EG experienced real-time adjustments of the DDA system, whereas the CG played a static, non-adaptive version.

Data Collection: Gameplay data (telemetry) and possession surveys (explicit feedback) were collected automatically for both groups as shown in table 3.

Table 4: Experimental and Control Group KPIs

KPI Category	Metric	Experiment al Group (DDA)	Control Group (Static)
Quantative	Deaths	Variance(E G_Deaths) vs. Variance(C G_Deaths)	Mean(EG_Death s) vs. Mean(CG_Death s)
	Time to Kill (TTK)	Variance(E G_TTK) vs. Variance(C G_TTK)	Mean(EG_TTK) vs. Mean(CG_TTK)
	Damag e Dealt/ Taken	Variance(E G_Damage) vs. Variance(C G_Damage)	Mean(EG_Dama ge) vs. Mean(CG_Dama ge)
	Avg. Velocit y	Correlate with Enjoyment/ Frustration	No DDA applied
Qualitative	Percei ved Difficu lty	Comparison of mean ratings	Comparison of mean ratings
	Percei ved Enjoy ment	Comparison of mean ratings	Comparison of mean ratings
	Player Retenti on	Proportion who would play again	Proportion who would play again

Analysis Plan

The collected data will be analyzed using statistical methods to compare the performance and satisfaction of the two groups. An independent samples t-test will be used to compare the means of quantitative KPIs, such as death count and TTK, between the EG and CG to determine whether the DDA had a statistically significant effect. Correlational analysis will be used within the EG to determine whether the implicit metrics (e.g., velocity fluctuation) correlate with explicit feedback (e.g., perceived frustration). Descriptive statistics and density plots will also be used to visualize the distribution of metrics across both groups, providing a clear picture of the DDA system's impact on player behavior and experience.

Anticipated Research Outcomes

If this experimental design were to be implemented in future research, the following outcomes are anticipated based on the theoretical framework: It is anticipated that the experimental group, which is exposed to the DDA system, will exhibit a more consistent performance profile than the control group. This is evidenced by a lower variance in metrics such as Time to Kill and death count. The DDA group was also expected to report higher levels of enjoyment and satisfaction on average, demonstrating the system's ability to maintain flow more effectively than a static difficulty model.

5. Theoretical Framework Validation and Analysis

Framework Design Strengths

The proposed DDA system addresses several key limitations identified in current adaptive difficulty approaches:

Novel Metrics Integration: The use of player velocity as a primary psychological indicator represents an innovative approach to implicit player state detection. Unlike traditional metrics that focus on discrete events (deaths, damage), velocity provides continuous behavioral data that may reveal player emotional states before they manifest in performance outcomes.

Hybrid Feedback Architecture: The integration of implicit telemetry with explicit player surveys creates a self-correcting system that can learn from both objective performance and subjective experience. This dual feedback approach addresses the fundamental challenge in DDA of optimizing for player satisfaction rather than purely statistical outcomes.

Technical Feasibility: The UE5 implementation leverages existing engine capabilities (Character Movement Component, Behavior Trees, Blueprint/C++ integration) without requiring extensive engine modifications, making the system practically implementable.

Theoretical Performance Predictions

Based on flow theory and existing DDA research, the proposed system should exhibit several characteristics:

Reduced Performance Variance: By continuously monitoring and adjusting to player state, the system should theoretically reduce the extreme outcomes (excessive deaths or trivial completion times) common in static difficulty systems.

Maintained Challenge Perception: The subtle nature of parameter adjustments (attack timing, cooldowns) rather than obvious changes (enemy health reduction) should preserve the player's sense of facing a consistent challenge.

Improved Engagement: The hybrid feedback loop should theoretically result in higher reported enjoyment and sense of control compared to static systems.

System Limitations and Considerations

Single-Player Scope: The current framework focuses exclusively on single-player encounters. Multiplayer environments introduce significant complexity in data collection, consensus-building among multiple player states, and fair adjustment application [15-16].

Velocity Assumption: The core assumption that player velocity patterns reliably indicate psychological states requires empirical validation. Individual player movement preferences may create noise in this signal.

Computational Overhead: Continuous velocity monitoring and real-time heuristic calculations may impact game performance, particularly in complex scenes with multiple dynamic elements.

Implementation Feasibility Analysis

The technical architecture demonstrates several strengths: **Modular Design:** The DDA Manager as an Actor Component allows for easy integration into existing projects without fundamental architecture changes.

Designer Accessibility: The hybrid C++/Blueprint approach enables rapid iteration by designers while maintaining performance-critical calculations in compiled code.

Scalability: The weighted heuristic approach allows for expansion to additional metrics without requiring algorithmic restructuring.

6. Discussion and Conclusion

Discussion of Findings

The framework suggests that player velocity could function as a valuable implicit metric, offering real-time insights into a player's psychological state that traditional combat metrics fail to capture. This theory predicts the potential of velocity-based measures to bridge the gap between surface-level performance indicators and deeper, more nuanced player experiences. If validated through empirical research, we anticipate that the hybrid system's ability to self-correct by incorporating explicit player feedback may prove superior to purely statistical models. The

model predicts that such an approach would not only enhance adaptive balancing in boss fights but also ensure the system prioritizes genuine player satisfaction rather than relying solely on maintaining a balanced win rate. Players would likely report a more engaging and rewarding gameplay experience under such conditions.

Limitations and Future Work

The primary limitation of this framework lies in its focus on a single-player boss-fight scenario. In multiplayer environments, network latency and the actions of multiple players would likely introduce significant complexity in data collection and application. The framework suggests that future research should explore how to scale this DDA system to a multiplayer context, potentially by employing a central, server-side DDA Manager to govern a raid boss's behavior in response to the collective performance of a player team.

While the current weighted heuristic algorithm appears powerful, the model predicts that it remains a limited approach. We anticipate that replacing the heuristic with machine learning models, such as Deep Reinforcement Learning (DRL) or neural networks, could enable more autonomous and sophisticated adaptation, allowing the system to learn and adjust across a wider range of gameplay scenarios without the need for manually defined weights.

Conclusion

The proposed framework can be seen as a potential step forward in the field of DDA design for climactic game encounters. By integrating a novel implicit metric (player velocity) with a robust explicit feedback mechanism, the theory predicts that it could provide a more comprehensive and adaptable solution for creating engaging, fair, and memorable boss fights in UE5. We anticipate that this hybrid approach offers a path for developers to design truly personalized game experiences that keep players in the elusive state of flow. This framework suggests that a smarter game need not be a 'scripted' one, but rather one designed to understand its player, fostering a deeper, more meaningful connection that extends beyond a single encounter.

REFERENCES

Annisa Damastuti, F., Dutono, T., Firmansyah, K., Barakbah, A., Hariadi, M., & Miftachul Arif, Y. (2024). Dynamic Level of Difficulties Using Q-Learning and Fuzzy Logic. IEEE Access, 12, 137775–137789. https://doi.org/10.1109/access.2024.3457801

Chowanda, A., Dennis, V., Dharmawan, V., & Ramli, J. D. (2023). Player's Affective States as Meta AI Design on Augmented Reality Games. JOIV International Journal on Informatics Visualization, 7(2), 561-568. https://doi.org/10.30630/joiv.7.2.1022

Constant, T., & Levieux, G. (2019). Dynamic Difficulty Adjustment Impact on Players' Confidence. In Proceedings of the 2019 CHI Conference on Human Factors in ComputingSystems,1–12. https://doi.org/10.1145/3290605.3300693

Frommel, J., Weber, M., Fischbach, F., & Rogers, K. (2018). Emotion-based Dynamic Difficulty Adjustment Using Parameterized Difficulty and Self-Reports of Emotion. In Proceedings of the 2018 International Conference on Advanced Visual Interfaces, 163–171. https://doi.org/10.1145/3242671.3242682

Fuchs, R., Gieseke, R., & Dockhorn, A. (2024). Personalized Dynamic Difficulty Adjustment – Imitation Learning Meets Reinforcement Learning. arXiv preprint arXiv:2408.06818.

Ishihara, M., Ito, S., Ishii, R., Thawonmas, R., & Harada, T. (2018). Monte-Carlo Tree Search for Implementation of Dynamic Difficulty Adjustment Fighting Game AIs Having Believable Behaviors. In 2018

IEEE Conference on Computational Intelligence and Games (CIG), 1–
8. https://doi.org/10.1109/cig.2018.8490376

Larsen, B. A., & Schoenau-Fog, H. (2019). "Well, That was Quick" – Towards Storyworld Adaptivity that Reacts to Players as People. In Interactive Storytelling, vol. 11869, pp. 204–213. https://doi.org/10.1007/978-3-030-33894-7_22

Limonad, L., & Azaria, A. (2017). EEG-Triggered Dynamic Difficulty Adjustment for Multiplayer Games. In Proceedings of the 22nd International Conference on Intelligent User Interfaces, 595-599. https://doi.org/10.1145/3025171.3025194

Saroğlu, Ö. C. (2024). Players Retell This Story. In Exploring the Future of Gaming and Gamification, pp. 133–158. https://doi.org/10.4018/979-8-3693-7235-7.ch006

Silva, M. P., Chaimowicz, L., & Do Nascimento Silva, V. (2015). Dynamic Difficulty Adjustment through an Adaptive AI. In 2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), 2, 173–182. https://doi.org/10.1109/sbgames.2015.16

Xue, S., Aghdaie, N., Kolen, J., Wu, M., & Zaman, K. A. (2017). Dynamic Difficulty Adjustment for Maximized Engagement in Digital Games. In Proceedings of the 26th International Conference on World Wide Web Companion,

465–471. https://doi.org/10.1145/3041021.3054170

Xu, L. (2024). The Effect of Game Experience on Flow Zone. Lecture Notes in Education Psychology and Public Media, 73(1), 135-140. https://doi.org/10.54254/2753-7048/73/20241051

Zhang, H., Agrawala, M., Sciutto, C., & Fatahalian, K. (2020). Vid2Player: Controllable Video Sprites that Behave and Appear like Professional Tennis Players. arXiv preprint arXiv:2008.04524. https://doi.org/10.48550/arxiv.2008.04524

Zheng, T. (2024). Dynamic difficulty adjustment using deep reinforcement learning: A review. Applied and Computational Engineering, 71(1), 157-162. https://doi.org/10.54254/2755-2721/71/20241633

Zohaib, M. (2018). Dynamic Difficulty Adjustment (DDA) in Computer Games: A Review. Advances in Human-Computer Interaction, 2018, 1–12. https://doi.org/10.1155/2018/5681652

Biemer, C. F. (2023). Dynamic Difficulty Adjustment via Procedural Level Generation Guided by a Markov Decision Process for Platformers and Roguelikes. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 19(1), 436-439. https://doi.org/10.1609/aiide.v19i1.27540

Huber, T., Mertes, S., Rangelova, S., Flutura, S., & André, E. (2021). Dynamic Difficulty Adjustment in Virtual Reality Exergames through Experience-driven Procedural Content Generation. 2021 IEEE Symposium Series on Computational Intelligence(SSCI),18.(https://doi.org/10.1109/SSCI50451.2021.9660 086)

Moon, J., Choi, Y., Park, T., Choi, J., Hong, J. H., & Kim, K. J. (2022). Diversifying dynamic difficulty adjustment agent by integrating player state models into Monte-Carlo tree search. Expert Systems with Applications, 205, 117677. https://doi.org/10.1016/j.eswa.2022.117677

Rahim, M., Moradi, H., Vahabie, A., & Kebriaei, H. (2023). Continuous Reinforcement Learning-based Dynamic Difficulty Adjustment in a Visual Working Memory Game. arXiv preprint arXiv:2308.12726. https://doi.org/10.48550/arXiv.2308.12726

Vang, C. (2022). The Impact of Dynamic Difficulty Adjustment on Player Experience in Video Games. Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal, 9(1), Article 7. https://doi.org/10.61366/2576-2176.1105