UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Autobalanceamento de jogos digitais a partir do uso de Controladores Proporcional-Integrativo-Derivativo (PID)

Cristiano Barroso Serra

Orientador

Tadeu Moreira de Classe

RIO DE JANEIRO, RJ - BRASIL
JANEIRO DE 2025

Autobalanceamento de jogos digitais a partir do uso de Controladores Proporcional-Integrativo-Derivativo (PID)

CRISTIANO BARROSO SERRA

DISSERTAÇÃO DE MESTRADO APRESENTADA COMO REQUISITO OBRIGATÓRIO ESTIPULADO PELO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA (PPGI) DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO). APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

gov.br Documento assinado digitalmente
Tadeu Moreira de Classe
Data: 29/01/2025 14:45:10-0300
Verifique em https://validar.iti.gov.br

Tadeu Moreira de Classe, D.Sc. — (UNIRIO)

gov.br Documento assinado digitalmente
Sean Wolfgand Matsui Siqueira
Data: 29/01/2025 15:17:52-0300
Verifique em https://validar.iti.gov.br

Sean Wolfgand Matsui Siqueira, D.Sc. — (UNIRIO)

gov.br Documento assinado digitalmente
ESTEBAN WALTER GONZALEZ CLUA
Data: 29/01/2025 15:22:34-0300
Verifique em https://validar.iti.gov.br

Esteban Walter Gonzalez Clua, D.Sc. — (UFF)

RIO DE JANEIRO, RJ - BRASIL
JANEIRO DE 2025

Catalogação informatizada pelo(a) autor(a)

S487 Serra, Cristiano Barroso
Autobalanceamento de jogos digitais a partir do uso de Controladores Proporcional-Integrativo-Derivativo (PID) / Cristiano Barroso Serra. -- Rio de Janeiro : UNIRIO, 2025.
85

Orientador: Tadeu Moreira de Classe.
Dissertação (Mestrado) - Universidade Federal do Estado do Rio de Janeiro, Programa de Pós-Graduação em Informática, 2025.

1. Jogos Digitais. 2. Autobalanceamento. 3. Proporcional-Integral-Derivativo (PID). I. Classe, Tadeu Moreira de, orient. II. Título.

Agradecimentos

A realização deste trabalho só foi possível graças ao apoio constante da minha família, meus pais, Sebastião e Maria do Carmo, à minha esposa Karla, às minhas filhas Alice e Isabela, e aos meus irmãos, Alexandre e Bruno, cuja presença e incentivo estiveram sempre presentes ao longo de toda a minha jornada acadêmica e pessoal. Agradeço imensamente o afeto, a compreensão e a motivação que todos me proporcionaram em cada etapa deste percurso.

Deixo minha sincera gratidão aos amigos que estiveram ao meu lado durante os desafios da pesquisa e aos colegas de trabalho que, com generosas contribuições, aconselhamentos e palavras de encorajamento, foram fundamentais para a conclusão deste trabalho.

Sou profundamente grato a todos os coordenadores e professores que, ao longo da minha trajetória, compartilharam seu conhecimento e me guiaram no caminho da busca contínua por excelência. Agradeço à UNIRIO, instituição que me proporcionou ensino de alta qualidade e um ambiente propício ao desenvolvimento acadêmico.

Por fim, expresso meu especial reconhecimento ao meu orientador, Professor Tadeu Moreira de Classe, pelo profissionalismo, empenho e dedicação. Sua orientação, combinada com o apoio e as críticas construtivas oferecidas, contribuíram de maneira inestimável para o amadurecimento das ideias que compõem este trabalho.

<Muito obrigado>,

Cristiano Barroso Serra.

SERRA, CRISTIANO BARROSO **Autobalanceamento de jogos digitais a partir do uso de Controladores Proporcional-Integrativo-Derivativo (PID)**. UNIRIO, 2024. 84 páginas. Dissertação de Mestrado. Programa de Pós-Graduação em Informática, UNIRIO.

RESUMO

Esta dissertação apresenta um estudo sobre autobalanceamento de jogos digitais, utilizando controladores Proporcional-Integrativo-Derivativo (PID) aliados a técnicas de *Game Analytics*. O objetivo geral é investigar de que forma o uso de controladores PID pode aprimorar a dificuldade de jogos de modo adaptativo.

Inicialmente, foi realizado um Mapeamento Sistemático da Literatura (MSL), permitindo identificar trabalhos que empregam técnicas variadas, como heurísticas, *machine learning* e algoritmos genéticos, em abordagens de *Dynamic Difficulty Adjustment* (DDA). Constatou-se que não há estudos descrevendo a aplicação de controladores PID em autobalanceamento. A proposta investigou o uso de controladores PID de duas formas: um PID remoto no jogo InfinityFire, com comunicação via servidor externo para processar variáveis e fornecer saídas em tempo real, e um PID local no jogo SpacePilot, que dispensou o servidor e armazenou dados via Google Sheets. Em ambos os casos, o controle ajustava variáveis do jogo conforme o desvio entre pontuação e *setpoint*, enquanto o *Game Analytics* facilitava a análise de métricas. No estudo exploratório com InfinityFire, o PID remoto demonstrou reações dinâmicas em pontuações acima do *setpoint*, conforme análises estatísticas. No estudo experimental com SpacePilot, o PID proporcionou maior distribuição de pontuações altas em comparação aos métodos Fácil e Linear, embora não houvesse diferenças significativas em acertos e erros.

Os resultados indicam que o controlador PID é uma alternativa robusta para balancear jogos, oferecendo respostas dinâmicas a variações de desempenho. Participantes apreciaram o equilíbrio de desafio proporcionado pelo PID em comparação a métodos tradicionais. As limitações incluem testes em pequena escala, protótipos simples e pouca diversidade de perfis. Futuramente, recomenda-se explorar jogos mais complexos, maior número de participantes e integrar o PID com aprendizado de máquina para aprimorar o autobalanceamento.

Palavras-chave: Jogos digitais, Autobalanceamento , Game Analytics , Proporcional-Integral-Derivativo(PID).

ABSTRACT

This dissertation presents a study on the auto-balancing of digital games using Proportional-Integral-Derivative (PID) controllers combined with Game Analytics techniques. The main objective is to investigate how the use of PID controllers can enhance game difficulty adaptively.

Initially, a Systematic Literature Mapping (SLM) was conducted to identify studies employing various techniques, such as heuristics, machine learning, and genetic algorithms, in Dynamic Difficulty Adjustment (DDA) approaches. It was found that no studies describe the application of PID controllers in auto-balancing. The study explored the use of PID controllers in two ways: a remote PID in the game InfinityFire, with communication via an external server to process variables and provide real-time outputs, and a local PID in the game SpacePilot, which eliminated the need for cloud communication and stored data via Google Sheets. In both cases, the control adjusted game variables based on the deviation between the player's score and the setpoint, while Game Analytics facilitated metric analysis. In the exploratory study with InfinityFire, the remote PID demonstrated dynamic responses in scores above the setpoint, according to statistical analyses. In the experimental study with SpacePilot, the PID provided a broader distribution of higher scores compared to the Easy and Linear methods, although no significant differences in hits and errors were observed.

The results indicate that the PID controller is a robust alternative for game balancing, offering dynamic responses to performance variations. Participants appreciated the balanced challenge provided by the PID compared to traditional methods. Limitations include small-scale tests, simple prototypes, and limited diversity of player profiles. Future studies are recommended to explore more complex games, a larger number of participants, and the integration of PID with machine learning to further refine auto-balancing.

Keywords: Digital games, Auto-balancing, Game Analytics, Proportional-Integral-Derivative (PID).

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Nomenclaturas

Autobalanceamento	Ajuste dinâmico da dificuldade de um jogo em função do desempenho do jogador.
Controlador PID	Controlador Proporcional-Integral-Derivativo.
DDA	Dynamic Difficulty Adjustment (Ajuste Dinâmico de Dificuldade).
Game Analytics	Área voltada à coleta e análise de dados de jogabilidade.
Integral (I)	Parte do controlador PID que acumula o erro ao longo do tempo.
Proporcional (P)	Parte do controlador PID que reage ao erro atual.
Setpoint	Valor de referência desejado (por exemplo, pontuação-alvo no jogo).
Spawn	Surgimento de elementos (inimigos, obstáculos) no jogo.

Lista de Símbolos e Unidades

SIGLA	UN.	SIGNIFICADO
PID	–	Controlador Proporcional-Integral-Derivativo
DDA	–	Dynamic Difficulty Adjustment (Ajuste Dinâmico de Dificuldade)
Kp	–	Ganho Proporcional
Ki	–	Ganho Integral
Kd	–	Ganho Derivativo
e(t)	–	Erro em função do tempo
MLS	–	Mapeamento Sistemático da Literatura (Systematic Mapping Study).
Δt	s (segundos)	Intervalo de tempo
v(t)	–	Velocidade do jogo em função do tempo
MSE	–	Mean Squared Error (Erro Quadrático Médio)

1. Introdução

O mercado de jogos eletrônicos atualmente configura-se como uma das indústrias mais robustas e dinâmicas do mundo, movimentando cifras significativas. A receita global da indústria de games ultrapassou os US$196 bilhões em 2022[1], e projeta-se um crescimento contínuo nos próximos anos, impulsionado pela crescente popularização de jogos móveis e o aumento da acessibilidade a plataformas de jogos (Smith; Doe, 2023b). Esse setor já ultrapassa outras indústrias de entretenimento, como a cinematográfica e a musical, consolidando-se como uma área de grande relevância econômica e cultural (Smith; Doe, 2023a; Johnson; Lee, 2022).

Apesar do expressivo crescimento econômico, a quantidade de estudos acadêmicos que explorem novas tecnologias relacionadas ao engajamento e adaptabilidade dos jogadores, bem como à coleta e análise de dados comportamentais dentro dos jogos, ainda é relativamente limitada (Serra; Classe, 2023; Šlosar *et al.*, 2022).

1.1 Problematização

Apesar do expressivo crescimento econômico e da popularização dos jogos eletrônicos, observa-se uma lacuna significativa na literatura acadêmica no que tange ao desenvolvimento e aplicação de tecnologias que potencializem o engajamento dos jogadores. A eficácia do engajamento, medida pelo tempo de envolvimento e pela retenção dos usuários, é crucial para o sucesso de jogos digitais, que visam muitas vezes objetivos além do entretenimento, como a educação, treinamento profissional ou promoção de comportamentos saudáveis (Miller; Thompson, 2020). No entanto, a escassez de estudos que investiguem métodos inovadores de coleta e análise de dados comportamentais dentro desses jogos limita a compreensão de como essas informações podem ser utilizadas para melhorar a experiência do jogador (Garcia; Pereira, 2021).

Além disso, a implementação de técnicas de autobalanceamento, que ajustam dinamicamente a dificuldade do jogo conforme as habilidades e o progresso do jogador, ainda enfrenta desafios técnicos relacionados aos tipos de balanceamento mais adequados para cada tipo de jogo (Kim; Lee, 2022). A falta de pesquisas aprofundadas sobre os algoritmos mais eficazes para o autobalanceamento e sobre as implicações

[1] <https://newzoo.com/resources/blog/the-games-market-will-show-strong-resilence-in-2022>

éticas e práticas da utilização de dados em tempo real impede o desenvolvimento de soluções robustas e responsáveis (Serra; Classe, 2023).

Portanto, baseado nessa lacuna, surge a necessidade de investigar como a combinação de coleta de dados comportamentais e técnicas de autobalanceamento pode ser otimizada para maximizar o engajamento dos jogadores em jogos digitais.

1.2 Justificativa

A justificativa é que novas abordagens de balanceamento podem permitir uma avaliação mais precisa da eficácia do autobalanceamento. A incorporação de tecnologias de informação avançadas, como os sistemas de autobalanceamento e as ferramentas de *Game Analytics*, em programas jogos digitais representa um avanço significativo na otimização do aprendizado e na maximização do engajamento dos participantes (Johnson; Wang, 2021).

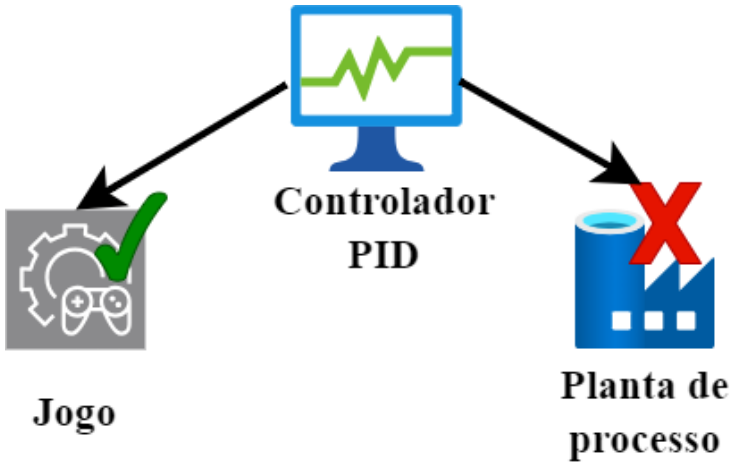
Entre as diversas possibilidades de integração dessas tecnologias, este estudo abordará especificamente o uso de controladores PID como uma solução viável para o autobalanceamento. Os controladores PID são ferramentas digitais amplamente utilizadas na indústria para o controle de variáveis como pressão, nível e temperatura, operando com base em valores de referência (*setpoints*) para regular processos (Vilanova; Visioli, 2017; Vilar; Souza, 2019). Adaptando essa tecnologia para o contexto dos jogos, conforme ilustrado na Figura 1.1, os controladores PID serão aplicados para balancear elementos do jogo, como pontuação e velocidade, proporcionando uma experiência de jogo mais equilibrada e adaptativa.

Ao fornecer um ambiente de aprendizado dinâmico e adaptativo, que integra coleta de dados e adaptação em tempo real, os jogos digitais têm o potencial de engajar os participantes de maneira mais eficaz e promover uma retenção de conhecimento mais robusta e aplicável, como por exemplo, no contexto de jogos com propósito (Pfau *et al.*, 2020; Serra; Classe, 2023). Essa abordagem não só aumenta o interesse e a motivação dos jogadores, mas também assegura que o aprendizado adquirido seja mais duradouro.

Ao abordar as implicações associadas à coleta e utilização de dados em tempo real, a pesquisa assegura que as práticas recomendadas respeitem a privacidade e a autonomia dos jogadores, alinhando-se com as normas e expectativas atuais. Isso fortalece a confiança dos usuários e estabelece bases sólidas para o desenvolvimento responsável de tecnologias nom jogos digitais (Thomas; Müller, 2021).

A relevância deste estudo está na sua capacidade de oferecer soluções inovadoras que atendam às demandas crescentes por métodos de autobalanceamento mais eficazes e personalizados, ao mesmo tempo, em que contribui para o avanço do conhecimento acadêmico.

Figura 1.1: Utilização do PID em Jogos



Fonte: Do autor.

1.3 Objetivo

O principal objetivo desta dissertação é apresentar e propor o uso de controladores PID como uma alternativa para o balanceamento de dificuldade em jogos digitais. Com o PID, a jogabilidade pode ser adaptada ao perfil do jogador por meio do monitoramento em tempo real das variáveis do jogo, permitindo uma experiência mais personalizada.

Para atingir esse objetivo, a pesquisa será orientada pelas seguintes questões:

- **Integrar o autobalanceamento em jogos para proporcionar uma experiência mais adaptativa.**
- **Apresentar os dados de *gameplay* de forma que possibilitem o acompanhamento do desempenho e da pontuação dos jogadores.**

Além do objetivo principal, a pesquisa busca cumprir os seguintes objetivos secundários:

- **Implementar um jogo que utilize o método de autobalanceamento proposto:** Esse objetivo envolve medir o impacto do autobalanceamento por meio de sua

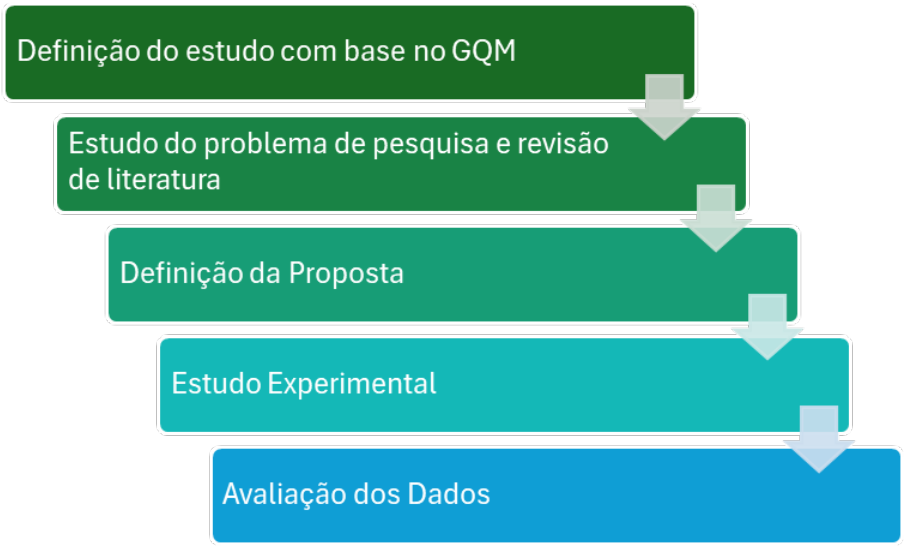
aplicação, avaliando se sua integração com o jogo resulta em melhorias estatisticamente significativas no desempenho e na percepção do jogador.

– **Comparar a aplicação do PID com métodos tradicionais (linear, estático) de autobalanceamento:** Esse objetivo visa verificar as diferenças e vantagens do uso de controladores PID em relação às técnicas tradicionais.

1.4 Metodologia

Este estudo foi desenvolvido em cinco etapas principais, que estão representadas no diagrama da Figura1.2. A seguir, cada uma dessas etapas é descrita de forma detalhada:

Figura 1.2: Diagrama das etapas do estudo.



Fonte: Do Autor.

1. **Estudo do Problema de Pesquisa e Revisão de Literatura:** Nesta etapa inicial, foi realizada a identificação do problema de pesquisa e uma revisão sistemática da literatura relevante. O objetivo foi compreender os conceitos, metodologias e soluções existentes, bem como identificar lacunas no conhecimento.

2. **Definição da Proposta:** Com base na revisão de literatura, foi formulada uma proposta para abordar o problema identificado. Essa proposta incluiu os objetivos específicos, as metodologias a serem empregadas e os recursos necessários para a execução do estudo.

3. **Estudo Experimental:** Nesta etapa, o estudo foi conduzido utilizando experimentos planejados para testar a proposta definida. Foram utilizados sistemas

simulados e/ou ambientes controlados para garantir a reprodutibilidade dos resultados.

4. **Avaliação dos Dados:** Os dados coletados nos experimentos foram analisados utilizando métodos estatísticos. Esta análise buscou verificar a eficiência e a validade da proposta.

5. **Resultados e Discussões:** Por fim, os resultados obtidos foram comparados com as soluções existentes e discutidos à luz da literatura revisada. Essa etapa incluiu a interpretação dos resultados, limitações do estudo e sugestões para trabalhos futuros.

1.5 Estrutura da Dissertação

No Capítulo 1, Introdução, são apresentados o contexto e a justificativa da pesquisa, destacando os desafios para métodos de autobalanceamento mais eficazes. São definidos os objetivos da pesquisa, a metodologia adotada e a estrutura da dissertação.

Capítulo 2: Conceitos Fundamentais, são discutidos os referenciais teóricos que sustentam a pesquisa, autobalanceamento, controladores PID e Game Analytics. A importância desses elementos no contexto de jogos digitais é enfatizada.

Capítulo 3: Trabalhos Relacionados, revisa a literatura existente, destacando os estudos que influenciaram a pesquisa e identificando lacunas que a dissertação pretende preencher.

Capítulo 4: Proposta de Solução, a solução desenvolvida é detalhada, incluindo o desenvolvimento do sistema, os macroprocessos envolvidos e a integração do controlador PID. A interação dos componentes e a adaptação com base nos dados coletados são explicadas.

Capítulo 5: Estudos Avaliativos, descreve a implementação e validação da metodologia proposta através de dois jogos - InfinityFire e SpacePilot. A problematização, coleta de dados e execução do jogo são abordadas, assim como as conclusões e limitações do experimento.

Capítulo 6: Considerações Finais, são resumidos os achados e as contribuições da pesquisa durante as atividades realizadas. As implicações e direções futuras são discutidas, sugerindo caminhos para pesquisas adicionais e desenvolvimento contínuo da solução proposta.

2. Conceitos Fundamentais

Este capítulo aborda os conceitos fundamentais que sustentam o desenvolvimento da dissertação. Inicialmente, explora a definição e a importância do autobalanceamento para personalizar a experiência dos jogadores conforme as habilidades. O capítulo também discute o uso de controladores PID, uma tecnologia essencial para o controle automático em processos industriais. Por fim, a seção sobre *Game Analytics* revela como a coleta e análise de dados podem otimizar a eficácia dos jogos digitais, oferecendo dados valiosos para o aprimoramento contínuo dos métodos de análise em jogos digitais. Ao integrar esses conceitos, o capítulo estabelece uma base teórica robusta para o desenvolvimento do experimento.

2.1 Autobalanceamento em Jogos

O *design* de jogos envolve a concepção da estrutura e o detalhamento das mecânicas, que incluem elementos lógicos e matemáticos essenciais como controle de fases, pontuação, valor de recompensas e curva de aprendizado. Essas mecânicas são fundamentais para garantir que o jogo não só funcione, mas que também seja envolvente e divertido. Um dos desafios mais significativos no *design* de jogos é o balanceamento, que visa ajustar essas mecânicas para alcançar uma experiência de jogo equilibrada e divertida (Xia; Anand, 2016).

O autobalanceamento é uma técnica de *design* de jogos que ajusta dinamicamente a dificuldade do jogo em resposta ao desempenho do jogador. Este processo visa manter o jogo desafiador e acessível, adequando-se às habilidades de cada jogador para maximizar a diversão e o engajamento. Esses ajustes podem ocorrer em diferentes aspectos do jogo: dificuldade dos desafios, alocação de recursos ou competência de oponentes controlados pela IA. Assim, o conteúdo se adapta à habilidade do jogador, buscando ampliar seu engajamento. (Altimira *et al.*, 2017; Schreiber, 2011).

O autobalanceamento pode envolver ajustes na dificuldade das tarefas, na alocação de recursos ou na competência dos adversários com base no desempenho dos jogadores. Por exemplo, se um jogador estiver enfrentando dificuldades excessivas, o jogo pode reduzir automaticamente a dificuldade dos desafios para torná-los mais gerenciáveis (Xia; Anand, 2016).

2.1.1 Estratégias de Autobalanceamento

Historicamente, o tipo de balanceamento mais tradicional na indústria de jogos é o **Balanceamento Estático**, pois se baseia em testes exaustivos (alfa e beta) e tende a permanecer inalterado após o lançamento, a menos que sejam aplicados *patches* ou atualizações (Reis, 2016; Hunicke *et al.*, 2004). A Tabela 2.1 estratifica os principais tipos de balanceamentos em jogos digitais.

Tabela 2.1: Estratégias de Autobalanceamento e Tipos de Balanceamento em Jogos

Estratégias de Autobalanceamento	
Heurístico	Utiliza regras e parâmetros pré-definidos (*heurísticas*) para adequar a dificuldade de acordo com o desempenho do jogador, ajustando fatores como força de inimigos ou recursos disponíveis. Discussões em (Altimira *et al.*, 2017; Andrade *et al.*, 2006).
Linear	Ajusta a dificuldade de forma diretamente proporcional a métricas simples (pontuação, número de vitórias/derrotas), seguindo incrementos fixos. Empregado em jogos com curva de aprendizado mais previsível (Yang *et al.*, 2017).
Por Inteligência Artificial	Utiliza algoritmos de aprendizado de máquina ou *reinforcement learning* para analisar padrões complexos em tempo real. A dificuldade é adaptada de modo sofisticado conforme os dados coletados sobre o jogador (Andrade *et al.*, 2006; Xia; Anand, 2016).
Tipos de Balanceamento	
Estático	Planejado antes do lançamento, depende de testes (alfa/beta) e permanece fixo depois da publicação, a menos que receba *patches*. É historicamente o tipo mais comum (Hunicke *et al.*, 2004; Reis, 2016).
Dinâmico	Ajusta aspectos do jogo em tempo real com base no desempenho dos jogadores, garantindo experiência equilibrada e envolvente. Frequentemente encontrado em jogos que buscam maior personalização (Xia; Anand, 2016; Schreiber, 2011).
Por Simetria	Garante condições iniciais idênticas para todos os jogadores, muito comum em jogos de estratégia e esportes eletrônicos. A vantagem competitiva resulta unicamente da habilidade (Reis, 2016).
Por Dificuldade e Habilidade	Adapta o jogo ao nível de habilidade presumido ou detectado do jogador, garantindo acessibilidade e desafio. Comum em jogos que oferecem modos *Easy*, *Normal* e *Hard* (Altimira *et al.*, 2017; Yang *et al.*, 2017).
De Ritmo	Regula a progressão de desafios, evitando sobrecarga ou tédio. Acompanha a curva de aprendizado do jogador ao longo do tempo (Hunicke *et al.*, 2004).

Fonte: Do Autor.

O autobalanceamento, quando aplicado a qualquer um desses tipos, permite uma calibragem contínua do nível de desafio, garantindo que jogadores com diferentes níveis de habilidade desfrutem de uma experiência justa e motivadora (Yang *et al.*, 2017; Altimira *et al.*, 2017).

2.2 *Game Analytics*

Game Analytics é um campo importante no desenvolvimento de jogos, focado na coleta, análise e interpretação de dados gerados pelas interações dos jogadores com o jogo. Este campo procura entender e otimizar a experiência do jogador, além de aprimorar a eficácia de jogos digitais. As informações derivadas de *Game Analytics* são essenciais para guiar os desenvolvedores na melhoria contínua do jogo (Midway, 2020; El-Nasr *et*

al., 2016).

Nos últimos anos, o interesse em *Game Analytics* cresceu substancialmente, impulsionado pelo aumento no volume de dados gerados por jogos online e pela necessidade de tomadas de decisão fundamentadas. Tais análises têm sido aplicadas a contextos diversos, desde jogos de pequeno porte até títulos AAA de grande orçamento (Lehmann *et al.*, 2023), exigindo pipelines robustos de coleta e processamento de dados para tomada de decisão ágil (Armstrong, 2021; Lee; Jin, 2021). Revisões sistemáticas apontam o crescimento das investigações sobre o comportamento de jogadores e o uso de dados de telemetria (Volkmar; Dahlskog, 2020), enquanto técnicas preditivas baseadas em aprendizado de máquina vêm sendo utilizadas para prevenir evasão (*churn*) e melhorar a retenção (Wang *et al.*, 2022).

A implementação de *Game Analytics* costuma começar com a extração de dados brutos diretamente das interações dos jogadores com o jogo. Esses dados são então processados e analisados para descobrir padrões ou tendências relevantes. A visualização, por meio de gráficos, mapas de calor ou tabelas, é uma etapa crítica, pois facilita a compreensão rápida de fenômenos complexos e ajuda a identificar áreas de sucesso ou pontos que necessitam de melhorias (El-Nasr *et al.*, 2016; Midway, 2020). Além das métricas tradicionais (pontuação, tempo de jogo, abandono), estudos recentes propõem indicadores mais abrangentes, visando capturar dimensões subjetivas da experiência do jogador (Cheung; Zimmerman, 2022) e destacar gargalos no *level design* por meio de análise de progressão (Feng; Burns, 2020).

A Tabela 2.2 consolida os principais tipos de *Game Analytics* descritos tanto na literatura clássica quanto em publicações recentes, cada um com suas características, objetivos e referências relacionadas.

Diante desse panorama, percebe-se que a área de *Game Analytics* vem se consolidando como peça-chave para o sucesso de títulos dos mais variados gêneros e tamanhos. A adoção de métodos avançados de análise de dados, *machine learning* e visualização interativa de resultados tem mostrado alto potencial de impactar positivamente a satisfação do jogador e a competitividade do jogo no mercado (Lehmann *et al.*, 2023; Lee; Jin, 2021).

2.2.1 Benefícios do *Game Analytics*

A implementação efetiva de *Game Analytics* oferece uma série de vantagens significativas (Su *et al.*, 2021):

Tabela 2.2: Principais tipos de *Game Analytics* e suas características

Tipo de Análise	Descrição e Referências
Análises de Comportamento de Jogador	Observam as ações dos jogadores (rotas, interações, etc.) para identificar pontos de abandono ou gargalos de *level design* (El-Nasr *et al.*, 2016; Feng; Burns, 2020). Relevantes em jogos online de médio e grande porte (Volkmar; Dahlskog, 2020).
Análises de Desempenho e Progressão	Avaliam métricas como pontuação, uso de recursos e vitórias/derrotas, identificando se a curva de aprendizado está adequada (Midway, 2020). Úteis para balanceamento e ajustes de dificuldade (Wang *et al.*, 2022).
Análises de Monetização	Direcionadas a entender comportamento de compra, uso de microtransações e assinaturas, auxiliando na definição de estratégias de precificação e ofertas (Midway, 2020; Armstrong, 2021).
Análises de Retenção e Engajamento	Estudam métricas como taxa de retorno, tempo médio de sessão e motivação de longo prazo. Auxiliam na identificação de pontos de desistência e no planejamento de conteúdo sazonal (El-Nasr *et al.*, 2016; Wang *et al.*, 2022).
Análises Sociais	Observam interações em clãs, trocas de itens e chats, investigando dinâmicas de comunidade e comportamento cooperativo ou competitivo (Midway, 2020). Têm sido foco em grandes títulos (*MMOs*, *e-sports*) (Volkmar; Dahlskog, 2020).
Análises Preditivas	Empregam algoritmos de aprendizado de máquina para prever *churn*, gastos futuros ou evolução de habilidades, permitindo intervenções de design e *marketing* direcionadas (Wang *et al.*, 2022; Armstrong, 2021).
Análises de UX e Usabilidade	Avaliam fatores de usabilidade (menus, clareza de objetivos) e satisfação geral. Podem combinar dados de telemetria com *eye tracking*, testes de usuário e questionários (Midway, 2020; Cheung; Zimmerman, 2022).

Fonte: Do Autor.

1. **Melhoria da Experiência do Jogador:** Ajustes fundamentados em dados detalhados podem aumentar substancialmente o engajamento e a satisfação dos jogadores, garantindo que cada aspecto do jogo seja ajustado para proporcionar a melhor experiência possível.

2. **Decisões Estratégicas:** As análises geram dados valiosos que orientam o desenvolvimento de futuras versões do jogo , assegurando que as decisões tomadas estejam bem informadas e sejam estrategicamente sólidas.

Em resumo, *Game Analytics* desempenha um papel fundamental na compreensão e no aprimoramento de softwares de jogos. Através da extração, tratamento, análise e visualização de dados, é possível obter uma compreensão valiosa que contribui para decisões mais precisas e eficientes, tornando os jogos mais atraentes e eficazes.

2.2.2 Processo de Uso de *Game Analytics*: Extração, Tratamento, Análise e Visualização de Dados

1. Extração de Dados (*Data Acquisition*): A extração de dados envolve a coleta de informações brutas diretamente das interações dos jogadores com o jogo. Esses dados podem incluir métricas de desempenho, ações do jogador, tempo gasto em diferentes níveis, decisões tomadas e muitas outras informações relevantes que descrevem a experiência e o comportamento do jogador (El-Nasr *et al.*, 2016). A extração precisa e

eficiente é fundamental para garantir que todos os aspectos críticos da *gameplay* sejam capturados.

2. Tratamento de Dados (*Data Cleaning*): Após a extração, os dados brutos frequentemente contêm inconsistências, imprecisões e erros que precisam ser corrigidos. O tratamento de dados é o processo de limpeza e preparação dos dados para análise. Isso inclui a remoção de duplicatas, correção de valores anômalos, preenchimento de dados ausentes e padronização de formatos. Este passo é crucial para garantir a precisão e integridade dos dados, que são essenciais para análises confiáveis (Midway, 2020).

3. Análise de Dados (*Data Analysis*): A análise de dados é a etapa onde os dados tratados são explorados e interpretados para gerar dados significativos. Existem três tipos principais de análises que podem ser aplicadas em *Game Analytics* (Roy *et al.*, 2022; Frutos-Pascual; Zapirain, 2015):

- **Análise Descritiva**: Fornece uma visão geral dos dados, destacando padrões gerais e estatísticas básicas, como médias, medianas e distribuições de frequência.

- **Análise Preditiva**: Utiliza modelos estatísticos e algoritmos de aprendizado de máquina para prever tendências futuras com base em dados históricos. Por exemplo, prever quais jogadores são mais propensos a abandonar o jogo.

- **Análise Prescritiva**: Vai além da previsão, sugerindo ações que podem ser tomadas para otimizar os resultados. Por exemplo, recomendações de ajustes de dificuldade para manter os jogadores engajados.

4. Visualização de Dados (*Data Visualization*): A visualização de dados envolve a representação gráfica das análises para facilitar a interpretação e comunicação dos resultados. Ferramentas de visualização podem incluir gráficos de barras, gráficos de linhas, diagramas de dispersão, mapas de calor e *dashboards* interativos. A visualização eficaz ajuda a identificar tendências, padrões e anomalias que podem não ser aparentes em dados brutos, permitindo uma tomada de decisão mais informada (El-Nasr *et al.*, 2016).

2.2.3 Aplicação de *Game Analytics* em Jogos

A aplicação de *Game Analytics* em jogos oferece uma oportunidade única para personalizar e otimizar a experiência de aprendizagem. Ao coletar e analisar dados gerados durante as sessões de jogo, desenvolvedores podem adaptar os elementos do jogo para melhor atender às necessidades individuais e habilidades dos jogadores.

Abaixo estão algumas das aplicações práticas mais impactantes de *Game Analytics* (Su *et al.*, 2021; El-Nasr *et al.*, 2016; Midway, 2020):

1. **Identificação de Padrões de Comportamento:** A análise detalhada de como diferentes jogadores interagem com o jogo pode revelar padrões comportamentais únicos. Esses padrões, uma vez identificados, podem ser usados para refinar o design do jogo . Por exemplo, se muitos jogadores falham repetidamente em uma determinada tarefa, isso pode indicar que a tarefa é demasiadamente complexa ou mal explicada.

2. ***Feedback* Personalizado:** Utilizando dados coletados durante o jogo, é possível oferecer *feedback* específico e personalizado aos jogadores. Esse feedback é baseado nas ações e no desempenho do jogador, fornecendo *insights* diretos que podem ajudá-los a compreender melhor suas falhas e sucessos, permitindo uma melhoria mais rápida e eficaz.

3. **Análise de Acertos x Erros:** Uma análise quantitativa dos acertos e erros dos jogadores durante as sessões de jogo fornece uma visão clara das áreas onde eles enfrentam mais dificuldades. Essa informação é vital para ajustar o nível de dificuldade do jogo ou para modificar as estratégias de ensino, garantindo que os jogadores sejam desafiados de maneira adequada sem se sentirem frustrados ou desmotivados.

Estas aplicações podem melhorar o engajamento do jogador. Ao integrar *Game Analytics* de forma eficiente, os jogos podem ser transformados em poderosas ferramentas de entretenimento ou educacionais que podem ser tanto motivadoras quanto informativas.

2.3 Controlador Proporcional-Integral-Derivativo

O controlador proporcional-integral-derivativo (PID) é uma ferramenta amplamente utilizada em sistemas de controle automático para manter variáveis em valores desejados. Este tipo de controlador é fundamental em diversas aplicações industriais e de engenharia devido à sua capacidade de proporcionar estabilidade e resposta rápida a perturbações (Aboelhassan *et al.*, 2020).

O PID recebe um sinal de entrada e retorna um sinal de saída para atuação. Por exemplo, um sinal do nível de água em uma caixa d'água é enviado como sinal de entrada,

esse sinal será processado pelo controlador PID e retornará um sinal de saída. Esse sinal de saída poderá ser usado para controle de abertura e fechamento de válvulas de água e/ou partida ou parada de bombas para controle do nível.

2.3.1 Componentes do PID

O controlador PID opera com base em três componentes principais: proporcional, integral e derivativo. Cada um desses componentes desempenha um papel específico na correção do erro entre a variável medida e o valor desejado (*setpoint*).

1. Componente Proporcional (P)

   O componente proporcional (P) responde à diferença atual entre a variável controlada e o valor desejado. Essa diferença é chamada de erro (*e(t)*). A ação proporcional ajusta a saída do controlador de maneira diretamente proporcional ao erro. A constante proporcional (*Kp*) determina a intensidade dessa resposta. A fórmula para o termo proporcional é:

   $$K_p e(t)$$

   onde:

   K_p é o ganho proporcional

   $e(t)$ é o erro no tempo t

2. Componente Integral (I)

   O componente integral (I) lida com a acumulação ao longo do tempo do erro (*e(t)*). Esse componente é responsável por corrigir erros residuais que não foram corrigidos pelo componente proporcional. A ação integral soma o erro ao longo do tempo e ajusta a saída do controlador para eliminar qualquer erro constante. A constante integral (*Ki*) determina a intensidade da resposta integral. A fórmula para o termo integral é:

   $$K_i \int_0^t e(\tau)\, d\tau$$

onde:

K_i é o ganho integral

$\int_0^t e(\tau)\,d\tau$ é a integral do erro ao longo do tempo

3. Componente Derivativo (D)

   O componente derivativo (D) considera a taxa de variação do erro ao longo do tempo. Ele prevê futuras tendências do erro e aplica correções para reduzir a velocidade com que o erro muda. A constante derivativa (*Kd*) determina a intensidade da resposta derivativa. A fórmula para o termo derivativo é:

$$K_d \frac{de(t)}{dt}$$

onde:

K_d é o ganho derivativo

$\frac{de(t)}{dt}$ é a derivada do erro em relação ao tempo

Fórmula Geral do PID

A combinação dos três componentes resulta na fórmula geral do controlador PID, que é usada para calcular a saída (*u(t)*) do controlador:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{de(t)}{dt} \tag{2.1}$$

onde:

$u(t)$ é a saída do controlador

$e(t)$ é o erro no tempo t

K_p é o ganho proporcional

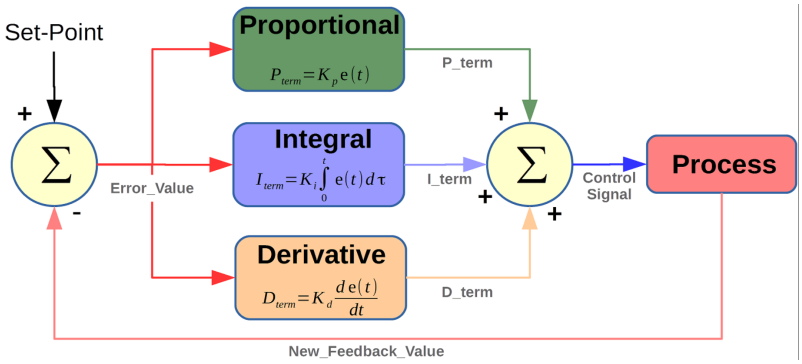
K_i é o ganho integral

K_d é o ganho derivativo

2.3.2 Aplicação do PID

Esses três elementos são combinados para gerar um sinal de controle que ajusta o sistema para minimizar o erro entre a variável medida e a referência desejada. O PID é flexível e eficaz em uma variedade de aplicações, proporcionando estabilidade e resposta rápida a perturbações. Ele é amplamente empregado em setores como automação industrial, controle de processos e robótica (Ariyansyah; Ma'arif, 2023).

Figura 2.1: Diagrama PID[1]



Fonte: The Engineering Concepts .

2.3.3 Benefícios do Uso do PID

O uso de um controlador PID traz diversos benefícios para os sistemas de controle automático: (Vilanova; Visioli, 2017)

- Estabilidade: Ajuda a manter a estabilidade do sistema, mesmo em presença de perturbações.
- Precisão: A combinação de ações proporcional, integral e derivativa permite corrigir erros de maneira precisa.
- Adaptabilidade: Pode ser ajustado para diferentes tipos de sistemas, tornando-o versátil para diversas aplicações.

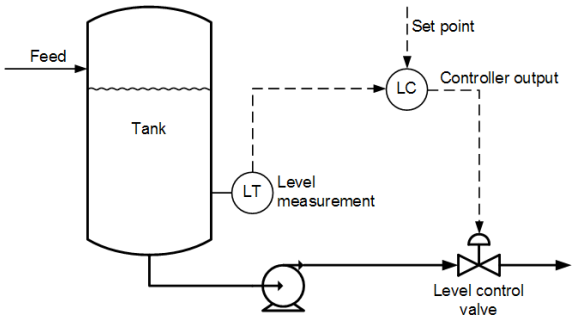
2.3.4 Exemplo Prático

Considere um sistema de controle de nível de um tanque, onde o nível desejado de líquido é o *setpoint*. O controlador PID ajusta a válvula de entrada para garantir que o

[1] <https://www.theengineeringconcepts.com/wp-content/uploads/2018/11/PID-CONTROLLER-BLOCK-DIAGRAM.png>

nível medido do líquido no tanque se mantenha próximo ao *setpoint*, respondendo rapidamente a qualquer variação no fluxo de entrada ou saída. A Figura 2.2 ilustra um sistema de controle PID aplicado ao controle de nível de um tanque. Este diagrama representa os componentes e o fluxo de informação e controle no sistema. Vamos detalhar cada parte do diagrama para entender como o controlador PID mantém o nível do tanque no valor desejado (*setpoint*).

Figura 2.2: Diagrama de Controle PID para Nível de Tanque[2]



Fonte: Blog Opticontrols.

1. **Tanque (*Tank*)**: Este é o tanque cujo nível de líquido precisa ser controlado. A entrada de líquido no tanque é mostrada pela seta com a etiqueta "*Feed*".

2. **Transmissor de Nível (LT - *Level Transmitter*)**: O transmissor de nível mede continuamente o nível do líquido no tanque. Esta medida é crucial para determinar a diferença entre o nível real e o nível desejado. O transmissor envia a informação de nível para o controlador de nível (LC).

3. **Controlador de Nível (LC - *Level Controller*)**: O controlador de nível recebe a leitura do nível do transmissor de nível (LT) e a compara com o nível desejado (*setpoint*). A diferença entre o nível atual e o *setpoint* é o erro ($e(t)$). Com base nesse erro, o controlador PID calcula a ação de controle necessária. O controlador então gera um sinal de saída que ajusta a válvula de controle de nível.

4. ***Setpoint***: O *setpoint* é o nível desejado do líquido no tanque. Este valor é definido pelo operador ou pelo sistema de controle para manter o nível dentro de um limite especificado.

5. **Saída do Controlador (*Controller Output*)**: Esta é a ação de controle calculada pelo controlador PID com base no erro. O sinal de saída é enviado para a válvula de controle de nível.

[2]<https://blog.opticontrols.com/wp-content/uploads/2011/12/Level-Control.png>

6. **Válvula de Controle de Nível (*Level Control Valve*)**: A válvula de controle ajusta o fluxo de entrada ou saída do tanque para manter o nível do líquido no *setpoint*. A válvula é aberta ou fechada parcialmente conforme o sinal de controle recebido do controlador de nível.

2.3.5 Funcionamento do Sistema

Abaixo, serão apresentadas as etapas essenciais para o funcionamento de um sistema de controle de nível, desde a medição até o ajuste final:

1. **1. Medição do Nível**: - O transmissor de nível (LT) mede o nível do líquido no tanque e envia essa informação para o controlador de nível (LC).
2. **2. Cálculo do Erro**: - O controlador (LC) compara o nível medido com o *setpoint* e calcula o erro (*e(t)*).
3. **3. Ação do Controlador PID**: - Com base no erro, o controlador PID aplica a fórmula:
$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{de(t)}{dt}$$
- Onde: - $K_p e(t)$ é a correção proporcional. - $K_i \int_0^t e(\tau)\,d\tau$ é a correção integral. - $K_d \frac{de(t)}{dt}$ é a correção derivativa.
4. **4. Ajuste da Válvula**: - O sinal de controle $u(t)$ é enviado para a válvula de controle de nível. - A válvula ajusta a taxa de fluxo de entrada ou saída do tanque, corrigindo o nível do líquido conforme necessário.

2.3.5.1 Benefícios do Controle PID no Sistema de Nível

Abaixo estão os principais benefícios da utilização do PID em processos de controle.

1. **Estabilidade**: Mantém o nível do líquido estável, reduzindo oscilações.
2. **Precisão**: Corrige desvios do nível desejado de forma eficaz.
3. **Adaptabilidade**: Ajusta-se a diferentes condições operacionais, como variações na entrada de líquido ou demanda.

Este diagrama e explicação mostram como o controlador PID pode ser utilizado para manter o nível de líquido em um tanque estável, reagindo de forma dinâmica a mudanças no sistema e garantindo que o nível permaneça próximo ao *setpoint* desejado.

O controlador PID pode ser uma opção de ferramenta em muitos sistemas de controle automático devido à sua capacidade de integrar ações corretivas de maneira proporcional, integral e derivativa. Com ajustes adequados das constantes *Kp*, *Ki* e *Kd*, o PID pode ser configurado para otimizar a performance do sistema em diversas condições operacionais.

2.4 Considerações Finais do Capítulo

Este capítulo apresentou conceitos fundamentais sobre autobalanceamento em jogos, controladores PID e *Game Analytics*. O autobalanceamento é apresentado como uma ferramenta fundamental para o engajamento dos jogadores com métodos para personalizar a experiência dos jogadores conforme as habilidades individuais. A aplicação de controladores PID foi explorada como uma ferramenta eficaz para manter a estabilidade e precisão no controle de variáveis, adaptando-se rapidamente a mudanças e proporcionando uma estabilidade no controle de plantas de processo. O PID, com seus componentes proporcional, integral e derivativo, é fundamental para diversas aplicações industriais e de engenharia, oferecendo ajustes contínuos e precisos.

A importância do *Game Analytics* foi destacada pela sua capacidade de extrair, tratar, analisar e visualizar dados de jogos, permitindo uma personalização e otimização da experiência do jogador. A análise de dados de jogos fornece dados valiosos para desenvolvedores e treinadores, ajudando a identificar padrões de comportamento, oferecer *feedback* personalizado e tomar decisões estratégicas informadas.

3. Trabalhos Relacionados

O presente capítulo descreve o processo de Mapeamento Sistemático da Literatura (MSL) realizado para investigar e sintetizar as principais abordagens de autobalanceamento em jogos digitais. O protocolo do estudo, apresentado no Apêndice A 6.5, segue as recomendações de (Kitchenham; Charters, 2007), contemplando a formulação das questões de pesquisa, a definição da *string* de busca, a seleção criteriosa dos artigos e a extração dos dados relevantes.

3.1 Mapeamento Sistemático da Literatura

O MSL (Protocolo no Apêndice A 6.5) é uma abordagem metodológica utilizada para identificar, avaliar e sintetizar de forma sistemática as evidências disponíveis em um determinado campo de estudo. Essa técnica de revisão sistemática visa mapear e analisar as publicações científicas relevantes sobre um tema específico (Kitchenham; Charters, 2007).

Seguindo o planejamento (protocolo detalhado no Apêndice 6.5), foram realizadas buscas em fontes de dados de abril de 2024 a dezembro de 2024. No total, foram encontrados 1532 estudos. Desses, 572 foram excluídos por serem duplicados (37% dos estudos). Na etapa de seleção, dos 960 estudos restantes, 25 estudos foram selecionados na etapa final (1,6% dos estudos), como pode ser observado na Tabela 3.1.

Tabela 3.1: Relação de estudos em cada etapa do MSL

Base	**Busca**	**Etapa 1**		**Etapa 2**	
		Duplicados		**Seleção**	
		Removidos	**Restantes**	**Removidos**	**Aceitos**
ACM Digital Library	344	41	303	301	2
El Compendex	250	196	54	48	6
IEEE Digital Library	79	56	23	20	3
Scopus	285	208	77	74	3
Web of Science	177	18	159	151	8
Springer	397	53	344	341	3
Total	**1532**	**572**	**960**	**935**	**25**

Fonte: Do Autor.

Ao criar a *string* de busca, foram incluídos sinônimos das palavras-chave em inglês. Dessa maneira, a *string* de busca foi formulada como:

("Digital game")

AND

("DDA" OR "Dynamic Difficulty Adjustment")

A Tabela 3.2 apresenta os 25 estudos aceitos para análise. Nela é apresentado um código (ID), o qual será responsável pela identificação do artigo ao longo da apresentação dos resultados, o ano da publicação, o título do estudo e a pontuação de qualidade após a leitura completa do texto.

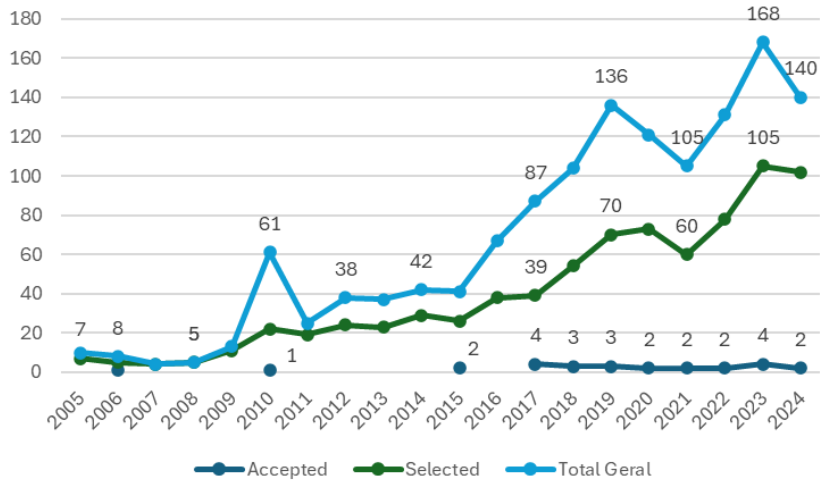
Tabela 3.2: Estudos aceitos

ID	Ano	Título	Qualidade
E-01	2015	A Data-driven Approach for Online Adaptation of Game Difficulty	6
E-02	2019	A Pilot Study on the Feasibility of Dynamic Difficulty Adjustment in Game-Based Learning	6
E-03	2015	Adaptation in Digital Games: The Effect of Challenge Adjustment	6
E-04	2006	Adaptive Game AI with Dynamic Scripting	6
E-05	2022	AI-enabled prediction of video game player performance	6
E-06	2018	Development and Assessment of an Adaptive Difficulty Arithmetic Game-Based Learning Object	6
E-07	2017	Dynamic Difficulty Adjustment for Maximized Engagement	6
E-08	2020	Dynamic Difficulty Adjustment in Digital Games Using Genetic Algorithms	6
E-09	2016	Dynamic Difficulty Adjustment on MOBA Games	6
E-10	2017	Dynamic Pressure Cycle Control: Dynamic Difficulty Adjustment	6
E-11	2018	Emotion-based Dynamic Difficulty Adjustment Using Parameterized Difficulty	6
E-12	2023	Fuzzy-based dynamic difficulty adjustment of an educational 3D-game	6
E-13	2021	Towards a Concept for a Hidden Object Game with Dynamic Difficulty Adjustment	6
E-14	2025	Extending a MAPE-K loop-based framework for Dynamic Difficulty Adjustment	6
E-15	2010	Micro-adaptivity: Protecting Immersion in Didactically Adaptive Digital Educational Games	6
E-16	2017	Visual Data Exploration for Balance Quantification During Exergaming	6
E-17	2019	δ-logit: Dynamic Difficulty Adjustment Using Few Data Points	6
E-18	2019	A Health Point-Based Dynamic Difficulty Adjustment Strategy for Video Games	6
E-19	2022	AlphaDDA: Strategies for Adjusting the Playing Strength of a Fully Trained AlphaZero System	6
E-20	2023	Behavioral and Psychophysiological Measures of Engagement During Dynamic Difficulty Adjustment in Immersive Virtual Reality	6
E-21	2024	Dynamic Difficulty Adaptation Based on Stress Detection for a Virtual Reality Video Game: A Pilot Study	6
E-22	2022	Improved Belgian AI Algorithm for Dynamic Management in Action Role-Playing Games	6
E-23	2022	Keep Calm and Aim for the Head: Biofeedback-Controlled Dynamic Difficulty Adjustment in a Horror Game	6
E-24	2017	Learning Constructive Primitives for Real-time Dynamic Difficulty Adjustment in Super Mario Bros	6
E-25	2020	User Evaluation of Affective Dynamic Difficulty Adjustment Based on Physiological Deep Learning	6

Fonte: Do Autor.

Conforme a Figura 3.1, a distribuição dos estudos ao longo dos anos mostra que os estudos sobre autobalanceamento de jogos começam a ser observados em 2005 e, a partir de 2008, houve um aumento significativo no número de estudos, indicando um crescente interesse voltado para o tema.

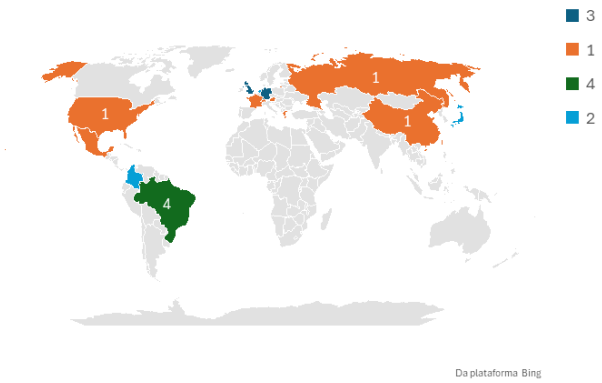
Figura 3.1: Distribuição de estudos por ano de publicação.



Fonte: Do autor.

O Brasil, conforme a Figura 3.2, tem a maior concentração de estudos, representando 45% do total, enquanto os demais, Áustria, Taiwan, Reino Unido, Singapura, Países Baixos e Estados Unidos, têm apenas 1 estudo cada, totalizando 9% cada. Essas concentrações podem indicar áreas de especialização, colaborações acadêmicas ou interesse em questões específicas relacionadas ao tema do estudo.

Figura 3.2: Distribuição geográfica dos estudos aceitos.



Fonte: Do autor.

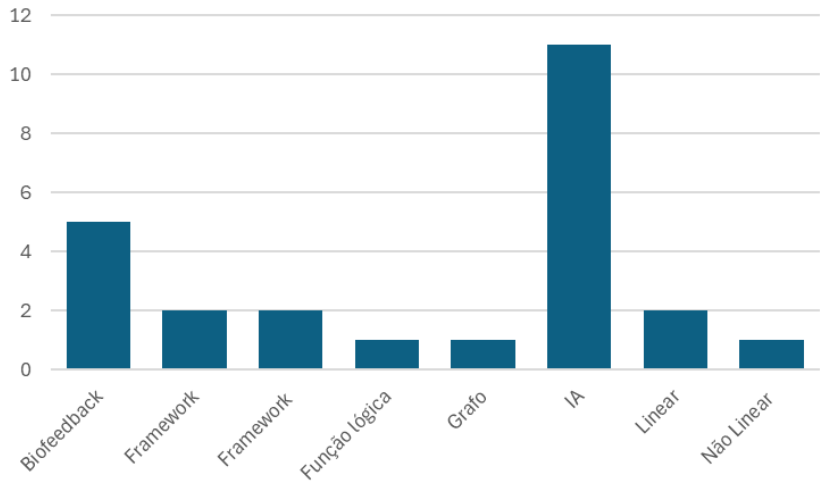
A distribuição da qualidade dos estudos selecionados apresenta uma pontuação elevada, 25 estudos com pontuação máxima de 6 pontos (100%), isso pode indicar que os resultados e conclusões obtidos a partir desses estudos são mais confiáveis e robustos.

3.1.1 Q1: Que técnicas de autobalanceamento são utilizadas em jogos digitais?

O objetivo desta questão é identificar as técnicas de autobalanceamento empregadas em jogos digitais quanto ao tipo de implementação. Dentre os 25 estudos examinados, destacam-se as seguintes categorias. Característica do tipo de algoritmo implementado:

- **IA/ML** (40% – 10 estudos) (E-01, E-04, E-05, E-08, E-12, E-17, E-19, E-22, E-24, E-25): uso de redes neurais artificiais, algoritmos genéticos, regressão logística, modelos de deep learning, entre outros.
- **Frameworks** (16% – 4 estudos) (E-13, E-14, E-15, E-16): utilização de *frameworks* que normalmente combinam mais de uma técnica de balanceamento de forma estruturada (ex.: MAPE-K loop-base, micro-adaptabilidade, Goal-based dda desing etc.).

Figura 3.3: Principais técnicas identificadas



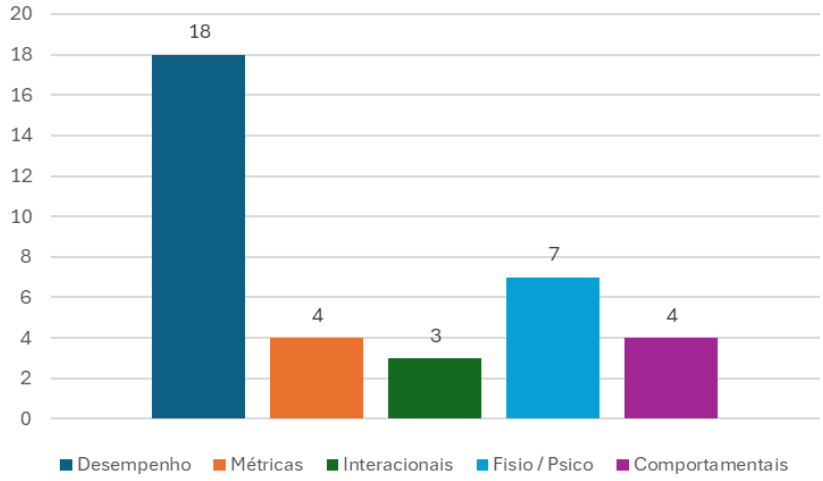
Fonte: Do autor.

3.1.2 Q2: Que tipo de dados são coletados para o autobalanceamento do jogo criado?

Essa questão tem como objetivo entender quais tipos de dados são coletados para o autobalanceamento de jogos digitais:

- **Dados de Desempenho** (72% – 18 estudos) (E-01, E-02, E-03, E-04, E-06, E-07, E-08, E-09, E-11, E-12, E-13, E-14, E-15, E-17, E-18, E-19, E-22, E-24): métricas que refletem a performance do jogador (tempo, acertos, erros, progressão etc.).

Figura 3.4: Tipos de dados coletados.



Fonte: Do autor.

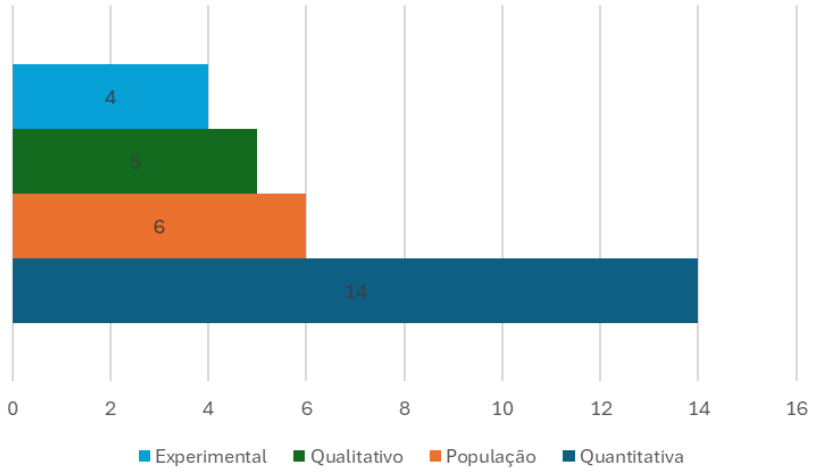
- **Dados Fisiológicos e Psicológicos** (28% – 7 estudos) (E-02, E-05, E-11, E-20, E-21, E-23, E-25): sinais de frequência cardíaca, estresse, emoções autorrelatadas, entre outros.
- **Dados Comportamentais** (20% – 5 estudos) (E-05, E-07, E-15, E-16, E-20): ações observáveis durante o jogo, como movimentação e tomadas de decisão.
- **Métricas de Jogo** (16% – 4 estudos) (E-01, E-10, E-12, E-19): informações específicas do jogo (nível ou fase, elementos de interface, quantidade de recursos etc.).
- **Dados Interacionais** (12% – 3 estudos) (E-01, E-02, E-12): interações do jogador com o sistema (cliques, toques na tela, uso de menus etc.).

3.1.3 Q3: Quais ferramentas foram utilizadas para validar a eficácia da utilização do autobalanceamento?

O objetivo dessa questão é saber quais ferramentas foram utilizadas para verificar a eficácia do autobalanceamento utilizado.

- **Métricas e Análises Quantitativas** (56% – 14 estudos) (E-02, E-03, E-05, E-07, E-08, E-11, E-12, E-15, E-16, E-17, E-20, E-21, E-22, E-25): testes estatísticos, correlações e métricas de desempenho.

Figura 3.5: Validação.



Fonte: Do autor.

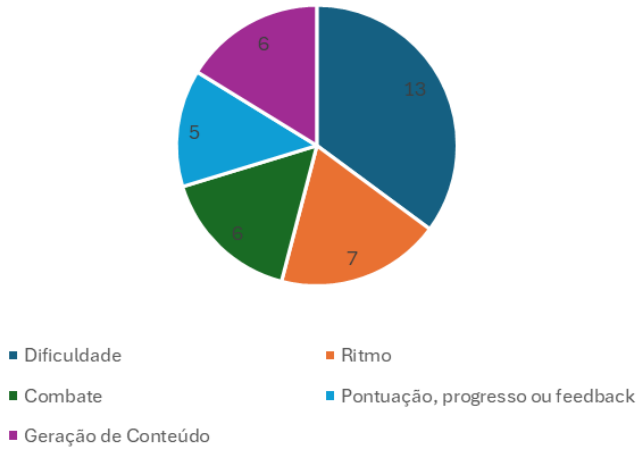
- **Populações de Participantes** (44% – 11 estudos) (E-06, E-09, E-10, E-11, E-12, E-13, E-17, E-18, E-22, E-23, E-25): coleta de dados e avaliação com grupos de usuários dos jogos.
- **Artefato Experimental** (24% – 6 estudos) (E-01, E-04, E-09, E-14, E-19, E-24): construção de experimentos controlados/protótipos para análise.
- **Instrumentos Qualitativos** (20% – 5 estudos) (E-02, E-03, E-06, E-12, E-23): questionários, entrevistas e relatos subjetivos.
- **Protocolos Experimentais** (16% – 4 estudos) (E-07, E-13, E-15, E-21): delineamentos experimentais com critérios e condições pré-definidas.

3.1.4 Q4: Quais as mecânicas do jogo criado que são influenciadas pelo balanceamento?

Essa questão tem o objetivo de mapear quais as principais mecânicas utilizadas para o autobalanceamento.

- **Ajuste e Modulação da Dificuldade** (52% – 13 estudos) (E-01, E-02, E-06, E-07, E-16, E-17, E-18, E-19, E-21, E-22, E-23, E-24, E-25): ajustes no nível de desafio, parâmetros de dificuldade, entre outros.
- **Ritmo, Velocidade e Intensidade** (28% – 7 estudos) (E-02, E-03, E-09, E-10, E-11, E-13, E-20): variação de velocidade de inimigos, frequência de obstáculos etc.

Figura 3.6: Mecânicas Influenciadas.



Fonte: Do autor.

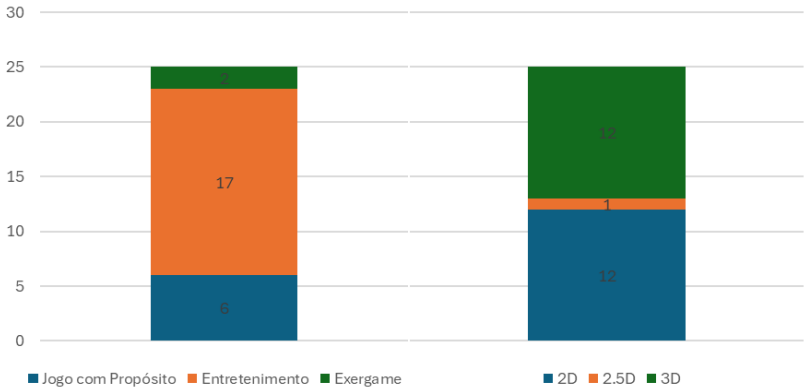
- **Mecânicas de Combate** (24% – 6 estudos) (E-04, E-09, E-12, E-14, E-18, E-22): mudança na IA de inimigos, dano causado, padrões de ataque.
- **Pontuação, Progresso e Feedback** (20% – 5 estudos) (E-03, E-05, E-07, E-14, E-15): uso de sistema de pontos ou feedback adaptativo.
- **Geração de Conteúdo e Ambiente** (24% – 6 estudos) (E-01, E-08, E-11, E-12, E-13, E-24): criação procedural de fases, posicionamento de itens.

3.1.5 Q5: Qual tipo de jogo foi criado?

Para essa questão, o objetivo foi mapear os tipos de jogos que implementam o autobalanceamento.

- **Jogos com Propósito** (24% – 6 estudos)
- **Jogos de Entretenimento** (68% – 17 estudos)
- **Exergames** (8% – 2 estudos) (E-16, E-20): focados em atividade física ou reabilitação.
- **Cenários 3D** (48% – 12 estudos) (E-01, E-02, E-03, E-04, E-05, E-06, E-12, E-15, E-16, E-20, E-21, E-23)
- **Cenários 2D** (48% – 12 estudos) (E-07, E-08, E-09, E-10, E-11, E-13, E-14, E-17, E-19, E-22, E-24, E-25)

Figura 3.7: Tipo de Jogos Utilizados.



Fonte: Do autor.

- **Cenário 2.5D** [1] (4% – 1 estudo) (E-18).

3.2 Conclusão do mapeamento

O mapeamento sistemático reuniu um total de 1532 artigos iniciais, culminando em 25 estudos aceitos após a aplicação dos critérios de seleção. A análise desses trabalhos evidenciou a diversidade e a complexidade das propostas de autobalanceamento em jogos digitais, contemplando tanto aspectos tecnológicos, como técnicas de IA/ML e frameworks de adaptabilidade, quanto diferentes tipos de dados, mecânicas de jogo e métodos de avaliação de eficácia.

No que diz respeito às técnicas de autobalanceamento (Q1), verificou-se uma predominância de algoritmos de Inteligência Artificial e Machine Learning (40%), enquanto 16% dos estudos recorreram a frameworks genéricos (por exemplo, MAPE-K), demonstrando interesse na padronização de processos adaptativos. Algumas pesquisas adotaram abordagens híbridas, combinando IA/ML com outros métodos (caso do estudo E-25, que integra Biofeedback). Em paralelo, o levantamento sobre os tipos de dados coletados (Q2) evidenciou um foco majoritário em métricas de desempenho (72%), mas também o crescimento no uso de dados fisiológicos e psicológicos (28%), enfatizando o interesse em adaptar a experiência do jogador não apenas pelo comportamento in-game, mas também por sinais biológicos e emocionais.

A avaliação da eficácia do autobalanceamento (Q3) mostra que 56% dos estudos

[1] Jogos 2.5D combinam gráficos 3D com jogabilidade restrita a um plano 2D, criando a ilusão de profundidade sem liberdade total de movimento no espaço tridimensional.

aplicam métricas e análises quantitativas, apontando para uma forte tendência de validação baseada em dados numéricos e estatísticos. Ainda assim, 20% dos artigos mencionam métodos qualitativos (como questionários e entrevistas), ressaltando a relevância de combinar abordagens objetivas e subjetivas. Em relação às mecânicas de jogo adaptadas (Q4), destaca-se o Ajuste e Modulação de Dificuldade (52%), seguido pelo controle de Ritmo, Velocidade e Intensidade (28%) e Mecânicas de Combate (24%), evidenciando como o equilíbrio dinâmico pode incidir desde aspectos gerais de progressão até sistemas de combate complexos.

Por fim, o mapeamento dos tipos de jogos (Q5) revelou que os estudos não se restringem a um único gênero ou finalidade, abrangendo jogos de entretenimento (68%), jogos com propósito (24%) e exergames (8%). Em termos de ambiente, há um equilíbrio entre cenários 3D (48%) e 2D (48%), além de menos pesquisas que explorem cenários 2.5D (4%). Tal variedade demonstra o potencial transversal do autobalanceamento em diferentes estilos de jogo, desde experiências sérias e educacionais até propostas puramente recreativas.

3.3 Considerações Finais do Capítulo

Ao final deste capítulo, observou-se que os estudos sobre autobalanceamento em jogos digitais cobrem uma ampla gama de cenários, desde jogos de entretenimento até aplicações sérias e exergames, empregando metodologias diversas para ajustar a dificuldade e manter o engajamento do jogador. Também se constatou uma tendência crescente no uso de técnicas de inteligência artificial, evidenciando o potencial de abordagem adaptativa. Apesar disso, o mapeamento mostra que ainda existem lacunas quanto à fundamentação matemática detalhada e métodos robustos de validação.

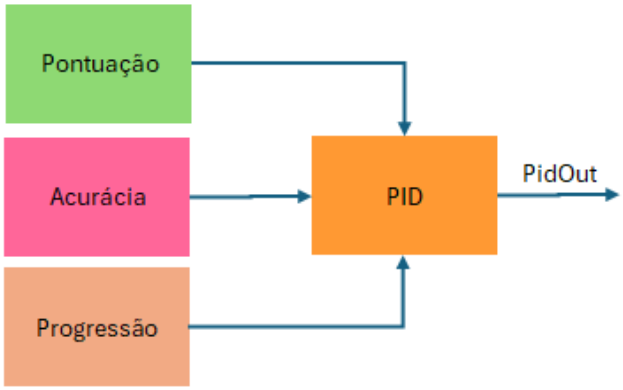
4. Proposta de solução

Este capítulo explora a implementação de controladores PID em dois jogos digitais distintos, visando ao autobalanceamento de dificuldade com base no desempenho do jogador. Na primeira parte, descreve-se o jogo **InfinityFire**, que faz uso de um PID remoto hospedado em servidor. Em seguida, aborda-se o jogo **SpacePilot**, que emprega um PID local na **Unity**.

Aplicação do PID nas propostas de solução:

Um controlador PID aplicado em jogos pode ser configurado para receber diversas entradas combinadas como pontuação, acurácia e progressão (Figura 4.1) e gerar uma única saída (*PidOut*). Essa abordagem unificada permite que o sistema ajuste a dificuldade do jogo de forma global. Por outro lado, é possível implementar controladores individuais para cada variável de entrada, proporcionando saídas individualizadas que podem oferecer um ajuste mais refinado do comportamento do jogo em diferentes variáveis para controle de dificuldade (Figura 4.2).

Figura 4.1: PID: Múltiplas variáveis com saída única.

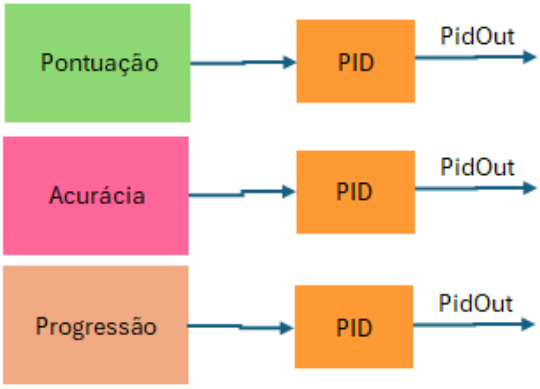


Fonte: Do Autor.

4.1 Implementação de Algoritmo de PID – Remoto

O sistema descrito implementa um controlador PID remoto, projetado para ajustar a dificuldade de um ambiente interativo com base no desempenho de um participante. Variáveis como *score*, *setpoint*, data, hora, ação executada e resultados (acerto ou erro)

Figura 4.2: PID: Múltiplas variáveis com saídas individualizadas.



Fonte: Do Autor.

são transmitidas ao servidor para análise e controle.

A Figura 4.3 ilustra o fluxo macro entre um usuário, o servidor e os *dashboards*. O sistema coleta e envia continuamente informações de desempenho ao servidor na nuvem, que processa o erro entre o *score* e o *setpoint* por meio do PID, gerando uma saída (pid_out).

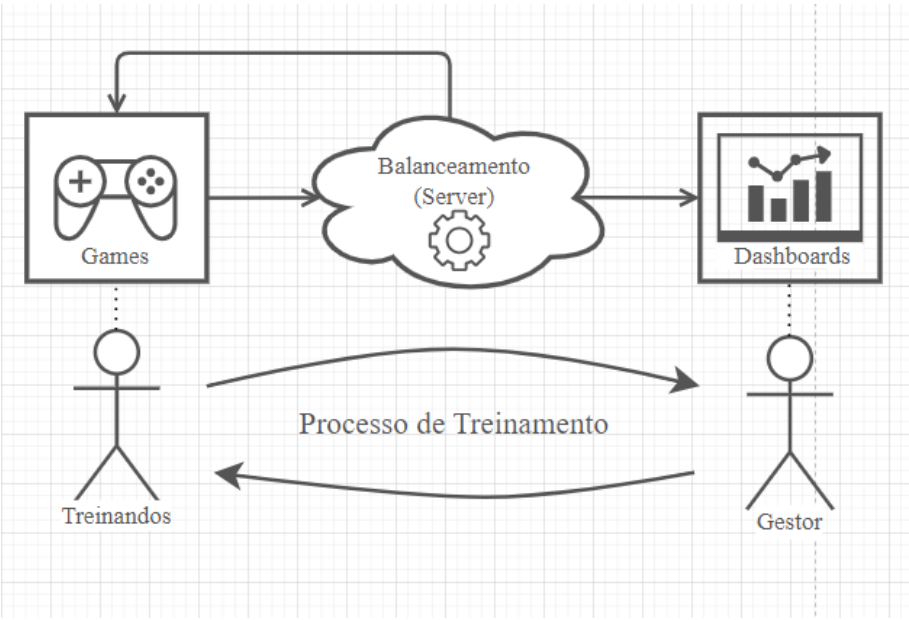


Figura 4.3: Fluxo de processos do sistema interativo

Fonte: Do Autor.

A Tabela 4.1 exemplifica o formato das requisições HTTP enviadas ao servidor, incluindo o nome da variável monitorada, seu valor atual e o *setpoint* desejado. A Tabela 4.2 apresenta a estrutura de resposta, que contém a saída do PID para reajustar a

dificuldade. A implementação completa do controlador, bem como das rotas de comunicação, está descrita no Apêndice A.

Tabela 4.1: Exemplo de requisição do InfinityFire ao servidor (PID remoto)

Campo	**Tipo**	**Descrição**
start	Boolean	Indica início da rodada
var_name	String	Variável monitorada (ex.: “score”)
new_value	Float	Valor atual do *score*
set_point	Float	Pontuação de referência
player	String	Identificador do jogador
date	String	Data e hora do registro
action	String	Ação executada
result	String	Resultado (acerto ou erro)

Fonte: Do Autor.

Tabela 4.2: Resposta típica do servidor, retornando a saída do PID

Campo	**Tipo**	**Descrição**
pid_out	Float	Saída do controlador PID
action_out	String	Ação sugerida (opcional)
player	String	Identificador do jogador

Fonte: Do Autor.

Na Figura **??**, é apresentada uma classe `PIDController` que gerencia o controlador PID, permitindo configurações persistentes em arquivo JSON. Essa classe oferece funcionalidades como carregamento de parâmetros, atualização de ganhos, cálculo do valor de saída e redefinição do estado interno.

Listing 4.1: Controlador PID Remoto (Pseudo-código)

```
CLASS PIDController:

    FUNCTION __init__(file_path, key):
        this.file_path = file_path
        this.key = key
        load_settings()

    FUNCTION load_settings():
        TRY:
            OPEN file_path
            PARSE settings AS JSON
            READ control_settings FOR key
```

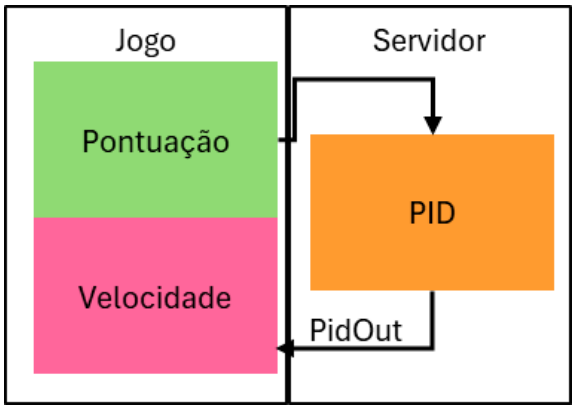
```
        CATCH file_not_found_error, json_decode_error:
            SET control_settings TO default
        SET kp, ki, kd, setpoint, prev_error, integral FROM control_settings

    FUNCTION save_settings():
        TRY:
            OPEN file_path
            PARSE settings AS JSON
        CATCH file_not_found_error, json_decode_error:
            SET settings = {}
        UPDATE settings[key] WITH kp, ki, kd, setpoint, prev_error, integral
        WRITE settings TO file_path

    FUNCTION update_k(kp, ki, kd):
        SET this.kp, this.ki, this.kd

    FUNCTION update_setpoint(setpoint):
        SET this.setpoint

    FUNCTION compute(current_value):
        error = setpoint - current_value
        integral += error
        LIMIT integral WITHIN [ -100, 100 ]  // Prevent integral windup
        derivative = error - prev_error
        output = (kp * error) + (ki * integral) + (kd * derivative)
        prev_error = error
        RETURN output

    FUNCTION get_integral():
        RETURN integral

    FUNCTION get_prev_error():
        RETURN prev_error
```

Fonte: Do Autor.

4.1.1 Demonstração do Jogo InfinityFire

O jogo InfinityFire simula o combate a incêndios, em que o jogador deve extinguir focos de chamas na zona verde para obter pontuação positiva. A Figura 4.4 exibe a tela do jogo, destacando os principais elementos: acertos e erros, o *Heads-Up Display* com o *timer* regressivo e o *score*, que representa a diferença entre as ações corretas e incorretas. Um incêndio do tipo sólido combustível (denominado Tipo_A) aparece circulado em amarelo na mesma figura. O objetivo final do jogador é atingir a melhor pontuação possível selecionando e debelando o princípio de incêndio com o extintor adequado antes que o tempo se esgote.

Figura 4.5: Controlador PID InfinityFire



Fonte: Do Autor.

Figura 4.4: Elementos principais do jogo InfinityFire

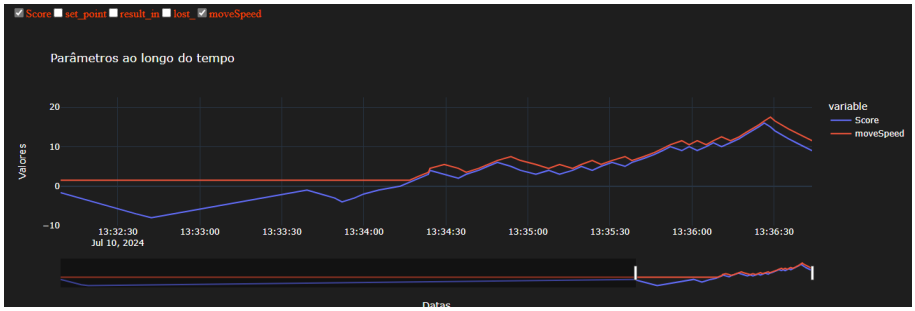
Fonte: Do Autor.

Durante a partida, o valor de pontuação (*score*) é transmitido em tempo real ao servidor para o cálculo do *PID Controller*, a entrada da pontuação no servidor gera um sinal de saída *PidOut*, esse sinal é utilizado para o balanceamento do jogo. Como observado na Figura 4.9, o sinal do *PidOut* será utilizado para balancear a velocidade do jogo.

A Figura 4.6 descreve a configuração e as características do *Dashboard* de Desempenho utilizado no experimento de treinamento baseado em jogos, a velocidade representada pela linha laranja e a pontuação em azul. Com esses dados, os gestores podem avaliar o desempenho do treinando ao longo do tempo e verificar os principais gargalos (desafios) no aprendizado:

O eixo X representa a data e hora, indicando o tempo de cada sessão de treinamento. Já o eixo Y exibe variáveis de desempenho relevantes para a análise. A primeira variável é o Score, que mostra a pontuação alcançada pelo jogador durante a sessão. A segunda é a **moveSpeed**, que indica a velocidade do jogo, ajustada dinamicamente pelo sistema. Por fim, há o **setPoint**, que ilustra o valor de pontuação desejado. Essas variáveis permitem uma análise detalhada do desempenho do jogador em relação ao tempo.

Figura 4.6: *Dashboard* de Desempenho



Fonte: Do Autor.

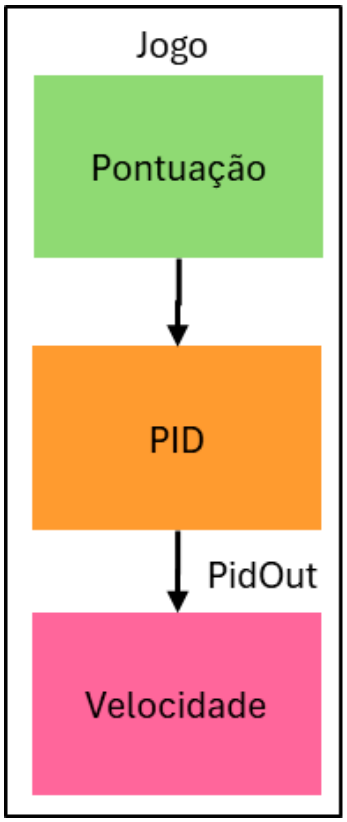
4.2 Implementação de Algoritmo de PID – Local

A biblioteca de balanceamento nativa no Unity foi utilizada para implementar o PID local. Dessa forma, o jogo não necessita de comunicação remota ou servidor em nuvem. A cada variação no *score*, o controlador interno calcula o erro em relação ao *setpoint*, aplica as parcelas proporcional, integral e derivativa e retorna um novo valor para o ajuste da velocidade.

Diferente do InfinityFire, o valor de pontuação (*score*) é transmitido em tempo real localmente em ambiente Unity para o cálculo do *PID Controller*, a entrada da pontuação é enviada para uma classe de autobalanceamento que utiliza a biblioteca Unity (**??**) criada para o projeto. Essa classe gera um sinal de saída *PidOut*, esse sinal é utilizado para o balanceamento do jogo. Como observado na Figura 4.7, o sinal do *PidOut* será utilizado para balancear a velocidade do jogo.

A biblioteca de autobalanceamento AutoBalanceLib (4.3) fornece uma implementação eficiente e modular para cálculos baseados no controlador PID (Proporcional-Integral-Derivativo). A classe principal, AutoBalanceController, encapsula os principais parâmetros do controlador, como os ganhos proporcional (Kp), integral (Ki) e derivativo (Kd), além do valor de referência (*Setpoint*) e do tempo de amostragem (Ts). Seu método principal, *Compute*, realiza o cálculo da saída do controlador em tempo real, considerando o erro entre o valor desejado e a variável de processo (pv). Internamente, a biblioteca integra o erro ao longo do tempo para o cálculo da componente integral, avalia a taxa de

Figura 4.7: Processo PID SpacePilot



Fonte: Do Autor.

variação para a componente derivativa e combina esses termos para gerar uma saída que pode ser utilizada em sistemas de controle. Além disso, a biblioteca inclui um método de reinicialização (*Reset*) que redefine as variáveis internas, garantindo flexibilidade para diferentes ciclos de operação.

Listing 4.2: Processo PID SpacePilot (Pseudo-código)

```
CLASS AutoBalanceController:

    // Parâmetros do controlador
    PUBLIC Kp, Ki, Kd, Setpoint, Ts
    PRIVATE integral, previousError

    FUNCTION CONSTRUCTOR (kp, ki, kd, setpoint, ts):
        Kp = kp
        Ki = ki
        Kd = kd
        Setpoint = setpoint
        Ts = ts
        integral = 0
        previousError = 0

    // Método para calcular a saída do controlador
    FUNCTION Compute(pv):
        error = Setpoint - pv
```

```
        integral = integral + (error * Ts)
        derivative = (error - previousError) / Ts
        output = (Kp * error) + (Ki * integral) + (Kd * derivative)
        previousError = error
        RETURN output

    // Método opcional para redefinir o controlador
    FUNCTION Reset():
        integral = 0
        previousError = 0
```

Fonte: Do autor.

Exemplo Implementação na Unity

A implementação do controlador PID 4.3 no *script* do Unity utiliza uma biblioteca de autobalanceamento, como demonstrado no código a seguir. Esse código pode ser utilizado em qualquer ambiente Unity, ajustando os parâmetros de entrada e as funções de integração para que o balanceamento possa ser realizado localmente.

Listing 4.3: Exemplo de controlador PID em Unity (Pseudo-código)

```
CLASS PIDController:

    PUBLIC Kp, Ki, Kd
    PRIVATE integral, lastError

    FUNCTION Update():
        setPoint = 10
        measuredValue = GetMeasuredValue()

        error = setPoint - measuredValue
        dt = tempoDecorrido() // Equivalente a Time.deltaTime
        integral = integral + (error * dt)
        derivative = (error - lastError) / dt

        output = (Kp * error) + (Ki * integral) + (Kd * derivative)
        ApplyOutput(output)

        lastError = error

    FUNCTION GetMeasuredValue():
        // Simula uma medição
        RETURN valorAleatorioEntre(8, 12)

    FUNCTION ApplyOutput(output):
        // Aplica o valor calculado
        Exibir("Saída do controlador: " + output)
```

Fonte: Fonte: Do Autor.

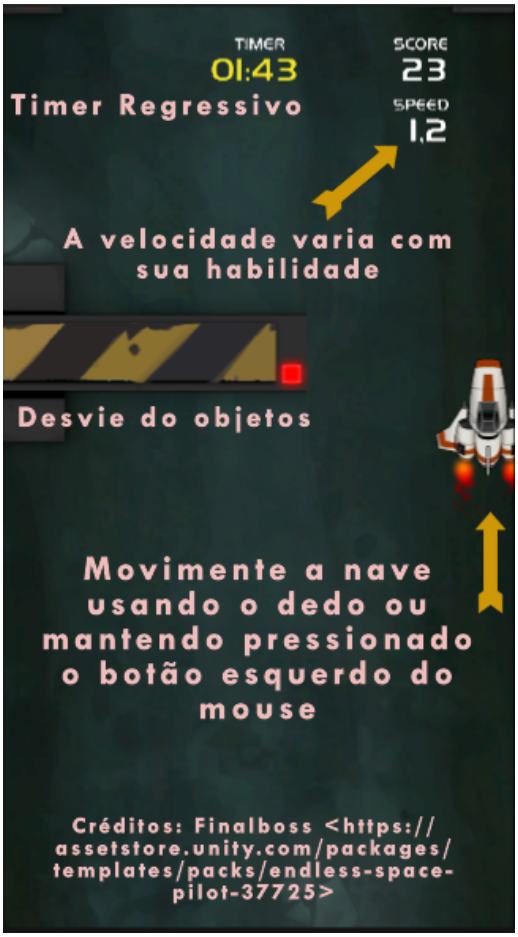
Este exemplo demonstra como integrar o controlador PID a um projeto Unity, ajustando dinamicamente o comportamento do jogo com base na diferença entre a variável de processo (*process variable*) e o valor desejado (*setpoint*). Durante a execução, o método

`Start` inicializa o controlador com os parâmetros configurados, enquanto o método `Update` é responsável por realizar os cálculos de correção a cada quadro. A saída do PID (`output`) pode ser aplicada diretamente ao jogo, como visto no código (**??**), ajustando variáveis como a velocidade ou a dificuldade. Embora o exemplo acima apenas exiba o valor no console, ele pode ser facilmente adaptado para enviar o `output` a um servidor ou interface de monitoramento.

4.2.1 Demonstração de Uso - SpacePilot (Local)

O jogo SpacePilot simula um cenário espacial, em que o jogador deve desviar uma nave de diversos obstáculos como portais e raios lasers. A Figura 4.8 exibe a tela do jogo, destacando os principais elementos: *Heads-Up Display* com o *timer* regressivo, velocidade (*speed*) e a pontuação *score*.

Figura 4.8: Jogo SpacePilot



Fonte: Do Autor.

O jogo SpacePilot foi adaptado para permitir a coleta de dados e o balanceamento dinâmico de dificuldade de forma local. Dessa maneira, o *score* do jogador e outras variáveis relevantes são enviados para a classe Unity AutoBal.cs (**??**) para cálculo e,

em seguida, esses dados são enviados por requisições `GET` para uma planilha no Google Sheets.

A partir do estudo com o InfintyFire, foi implementado no jogo SpacePilot diferentes níveis de *setpoints*. A partir de 20 pontos, a cada mais 20 pontos o *setpoint* é atualizado (20, 40, 60 e etc). No início do jogo, o PID inicia com *setpoint* de 20 pontos. Caso o jogador atinja 40 pontos, um novo *setpoint* será atribuído em 40 pontos e assim por diante; este ajuste foi implementado para oferecer novos desafios ao jogador e um maior dinamismo ao jogo.

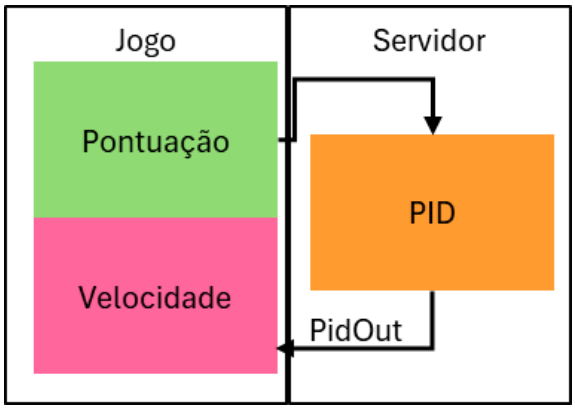
Na coleta de dados, o jogo envia informações como o nome do jogador, pontuação obtida, velocidade e tempo de partida restante. A análise desses registros pode ser realizada diretamente na planilha, tornando o processo de monitoramento mais simples e acessível. Na versão do SpacePilot, o envio das informações listadas na Tabela 4.3 ocorre a cada 5 segundos durante a execução da partida.

Listing 4.4: Implementação do AutoBal em SpacePilot

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using AutoBalanceLib;

namespace EndlessSpacePilot
{
    public class AutoBal : MonoBehaviour
    {
        public static AutoBalanceController controller;
        public static float setpoint = 20.0f; // Valor desejado
        public float kp = 0.03f;
        public float ki = 0.001f;
        public float kd = 0.03f;
        public static float processVariable = PlayerManager.playerScore; // Variável a
            ser coletada do jogo
        public static float output = 0;

        void Start()
        {
            controller = new AutoBalanceController(kp, ki, kd, setpoint, Time.deltaTime)
                ;
        }

        void Update()
        {
            if (PlayerNameCapture.isNameCap)
            {
                controller.Ts = Time.deltaTime;
                output = controller.Compute(PlayerManager.playerScore);
                print("-----------------Pidout " + output);
            }
        }
    }
}
```

Fonte: Do Autor.

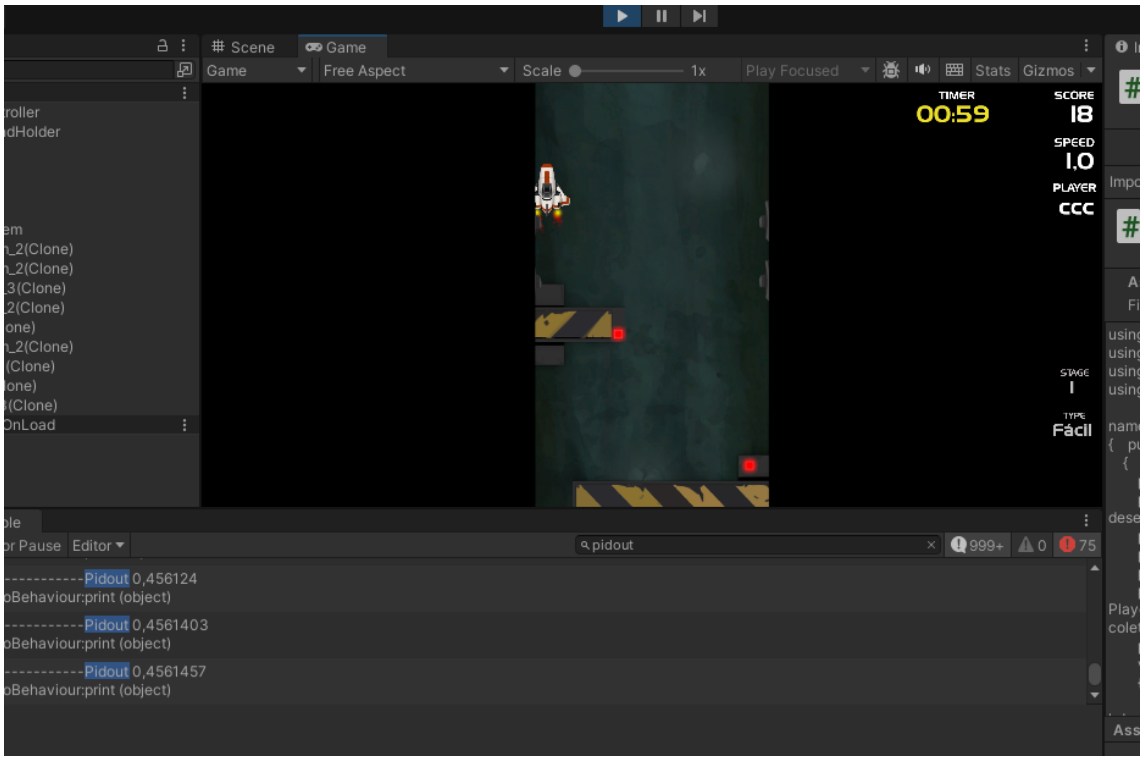
Figura 4.9: Controlador PID InfinityFire



Fonte: Do Autor.

Na Figura 4.10, observa-se a saída dos valores do PID calculado em tempo de execução do jogo. O *print* listado no console do *Unity* com o texto *Pidout* representa o valor de ajuste utilizado para o autobalanceamento.

Figura 4.10: Saída de PID durante execução



Fonte: Do Autor.

O processo de integração foi projetado de forma a ampliar a acessibilidade e a facilidade

de análise dos dados. Dessa forma, vários dispositivos (PC, *smartphones* ou tablets) podem se conectar online ao jogo, atualizar as informações de partida e receber o novo valor de velocidade (ou outro parâmetro de dificuldade) calculado pelo controlador PID local.

Tabela 4.3: Exemplo de campos enviados ao Google Sheets pelo SpacePilot remoto

Campo	Tipo	Descrição
Data e Hora	String	Data e horário do envio ou registro.
PlayerId	String	Identificador único do jogador .
Player	String	Nome ou apelido do jogador .
Score	Inteiro	Pontuação acumulada no jogo .
moveSpeed	Float	Velocidade atual aplicada ao jogo
Time	Inteiro	Tempo restante de partida, em segundos .
Difficult	Inteiro	Nível de dificuldade atual .
Tipo de Balanceamento	Inteiro	Indica o método de balanceamento empregado

Fonte: Do Autor.

Essa lista de campos pode ser adaptada conforme as necessidades de cada projeto, permitindo que desenvolvedores insiram novas variáveis ou removam aquelas que não são relevantes para a análise.

4.3 Considerações Finais

As abordagens apresentadas evidenciaram como o controlador PID pode ser empregado no autobalanceamento de jogos digitais, seja de modo remoto (InfinityFire) ou local (SpacePilot). No InfinityFire, o servidor em nuvem possibilita escalabilidade e análise em tempo real de múltiplos treinandos, armazenando dados em arquivos JSON. Já no SpacePilot, a implementação local na Unity reduz a complexidade de infraestrutura, contando com integração a uma planilha Google Sheets para coleta de métricas.

5. Estudos Avaliativos

Este capítulo descreve a metodologia adotada para o desenvolvimento e avaliação de um sistema de autobalanceamento para jogos digitais, utilizando controladores PID integrados com *Game Analytics*. Inicialmente, são apresentados os princípios metodológicos, fundamentados no modelo experimental de (Wohlin *et al.*, 2012), e as etapas de planejamento e execução do experimento. Em seguida, são detalhados dois estudos: o estudo exploratório com o jogo *InfinityFire*, que buscou validar os conceitos iniciais de autobalanceamento, e o estudo experimental com o jogo *SpacePilot*, que avaliou a eficácia de diferentes métodos de balanceamento em cenários variados. Os resultados obtidos são analisados tanto de forma quantitativa quanto qualitativa.

A metodologia adotada nesta pesquisa segue os princípios estabelecidos por (Wohlin *et al.*, 2012) em *Experimentation in Software Engineering*. O processo experimental é composto por cinco etapas principais: definição, planejamento, operação, análise e interpretação, e apresentação e empacotamento. O objetivo central é desenvolver e avaliar o autobalanceamento PID integrado com *analytics* em jogos digitais.

Conforme recomendado por Wohlin *et al.* (2012), a etapa de definição envolve o estabelecimento dos objetivos do experimento e a formulação de questões de pesquisa claras.

1. Identificação dos Objetivos:
2. Formulação de Questões de Pesquisa e Hipóteses:
3. Seleção do Contexto:

5.1 Estudo Exploratório – InfinityFire

5.1.1 Definição e Planejamento do Estudo

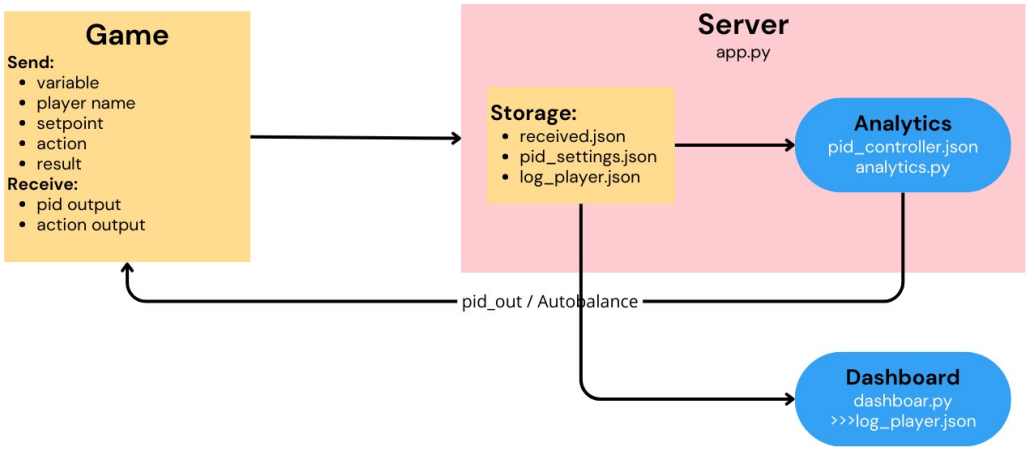
A definição do estudo apresenta os objetivos gerais e específicos da pesquisa. O objetivo (*goal*) pode ser descrito como: *investigar a viabilidade do balanceamento dinâmico de um jogo de treinamento por meio do controlador PID*; com o propósito de *otimizar a experiência de aprendizado e aprimorar o desempenho dos jogadores*; no que diz respeito a *identificar a influência das variáveis de jogo (velocidade e tempo de spawn de objetos) no desempenho do jogador (pontuação, acertos e erros)*; do ponto de vista de *jogadores envolvidos em de treinamentos baseados em jogos*, no contexto de *um experimento com uso de Game Analytics, servidor Python em nuvem e a engine Unity para criação do jogo*.

De acordo com o *modelo de avaliação* (inspirado em (Basili *et al.*, 1992)), as questões de pesquisa (*question*) a que se pretende responder são as seguintes:

- **Q1**: O balanceamento dinâmico via PID consegue ajustar de forma apropriada as variáveis de velocidade e tempo de *spawn* de objetos em função da pontuação do jogador?
- **Q2**: Qual é o impacto do balanceamento tradicional em comparação ao PID no desempenho (pontuação, acertos e erros) do jogador?

Para cada uma dessas perguntas, foram pensados indicadores (*metrics*) que auxiliarão na mensuração dos resultados obtidos.
Quando se investiga a *eficácia do ajuste PID* (Q1), busca-se avaliar se o controlador consegue estabilizar a pontuação em torno do valor desejado (no caso, **Score** = 6 pontos). Dessa forma, a métrica (**M1**) utilizada é *(M1) = número de sessões consideradas apropriadas pelos usuários*, considerando estabilizada a sessão cujo *Score* promova um jogo desafiador para o jogador.
No caso de *impacto comparativo entre o método tradicional e PID* (Q2), pretende-se observar o comportamento das variáveis *Speed_Trad* e *Speed_PID*, bem como *Score_Trad* e *Score_PID*. Assim, do ponto de vista quantitativo, pode-se adotar uma métrica semelhante, por exemplo, **(M2)** = verificação da correlação entre velocidade e pontuação para cada método. A distinção entre o tradicional e o PID ocorre por meio da análise estatística e gráfica (por exemplo, Figura 5.4 e Figura 5.3).
Para (M1) e (M2) , considera-se que a quantidade de *instâncias de dados* esperadas seja igual ao número de sessões de jogo registradas (*sessões executadas no Unity, com envio de dados para o servidor Python*). Essa definição baseia-se no *framework* apresentado na Figura 5.1, que coleta variáveis como *Score*, *Speed* e *spawnTime* a cada rodada de 5 minutos. Justifica-se esse valor, pois cada sessão gera uma amostragem completa das variáveis de desempenho. Estatisticamente, tais métricas podem ser tratadas na escala de *razão* (para calcular índices de estabilização, variação etc.), permitindo análises descritivas e inferenciais (por exemplo, correlações entre velocidade e pontuação).

Figura 5.1: Modelo Detalhado



Fonte: Do Autor.

Os participantes foram selecionados por conveniência, levando em consideração o acesso a indivíduos com perfil de **jogadores de nível iniciante, intermediário ou avançado**. Eles foram convidados a participar voluntariamente do estudo após a realização de um breve *tutorial* sobre o funcionamento básico do jogo, de forma que tivessem familiaridade mínima com as dinâmicas de *controle de incêndios* apresentadas na Figura 5.2. A escolha desse perfil justifica-se pela necessidade de observar como o sistema PID e o método tradicional se comportam em usuários com diferentes graus de experiência em jogos.

Para analisar *(1) o quão eficiente é o balanceamento PID* e *(2) o impacto do método tradicional* em diferentes contextos, optou-se por distribuir os participantes em dois grupos. O Grupo A utilizou **o método Tradicional** (com velocidade e tempo de *spawn* variando linearmente) e o Grupo B utilizou **o método PID** (com ajuste de velocidade e *spawn* determinados pelo controlador PID). Essa divisão permite investigar, em detalhes, como cada técnica de balanceamento responde a variações de *Score* e à curva de aprendizagem.

O estudo foi projetado para ocorrer em 2 fases principais, conforme a Tabela 5.1. A primeira etapa (E1) envolveu um *treinamento* (apresentação do método e leitura das instruções do jogo) para garantir que os participantes estivessem aptos a interagir com as variáveis do jogo. Já na segunda etapa (E2), houve a execução propriamente dita da aplicação de cada método (**Tradicional** ou **PID**).

Para apoiar a execução de cada etapa, foram disponibilizados aos participantes *tutorial* , exibindo informações sobre como utilizar o extintor na zona verde, conforme a Figura 5.2. Durante a fase E2, utilizou-se também um servidor **Python em nuvem**, responsável pelo *processamento de Game Analytics* e pelo ajuste PID em tempo real. O sistema registrou continuamente o *Score*, a velocidade atual do jogo (*Speed*) e o intervalo de *spawn* de objetos. Esses registros foram armazenados em um banco de dados para posterior análise,

Tabela 5.1: Etapas de Execução do Estudo

Etapa	Duração (min)	Descrição
E1	5	Treinamento, introdução ao método e leitura de instruções
E2	5	Aplicação prática do método (Tradicional ou PID)

Fonte: Do Autor.

servindo de base para o cálculo das métricas M1, M2 .

Antes da condução oficial do estudo, foi realizado um *estudo piloto* com 2 participantes para validar o planejamento, identificando possíveis falhas na coleta de dados e pontos de melhoria no tutorial apresentado aos jogadores. As sugestões e correções feitas pelos participantes do piloto foram incorporadas ao planejamento final, garantindo maior consistência metodológica e reduzindo o risco de viés ou dificuldades de entendimento durante as etapas do experimento.

A coleta de dados ocorreu de duas maneiras : (1) **Logs** gerados automaticamente com as informações de *Score*, velocidade e *spawnTime* a cada rodada de 5 minutos e (2) **artefatos resultantes da aplicação do jogo**, como os *dashboards* (Figura 4.6) que ilustram em tempo real o comportamento da pontuação e da velocidade;

Por fim, foram consideradas as possíveis ameaças à validade do estudo, tais como *ameaças de conclusão, internas, de construção e externas* (Creswell; Creswell, 2017). Na Tabela 5.2, constam as principais ameaças identificadas, bem como as estratégias adotadas para mitigá-las. Por exemplo, conduzir um treinamento inicial para os participantes reduz a **ameaça interna** (falta de familiaridade com o jogo), enquanto o uso de instrumentos testados no piloto e a avaliação estatística apropriada objetivam controlar a **ameaça de conclusão** (poder estatístico insuficiente).

A seguir na Tabela 5.3, detalha-se cada instância de dados enviada ao servidor em um sistema de treinamento baseado em jogos:

Tabela 5.2: Ameaças à Validade e Tratamentos

Tipo	**Ameaça**	**Descrição**	**Tratamento**
Conclusão	Poder estatístico	Falta de participantes pode afetar a robustez da análise	Uso de ferrementas matemáticas adequadas à amostas mínima
Interna	Falta de treinamento	Participantes podem não entender a mecânica do jogo e do controlador	Aula inicial, guia prático e etapa de treinamento (E1)
Construção	Instrumentação	Possibilidade de *bugs* ou erros de coleta de dados no servidor Python	Estudo piloto e revisão por pares do código
Externa	Generalização	Amostra limitada para extrapolar resultados	Replicação futura com amostras de diferentes perfis

Fonte: Do Autor.

Tabela 5.3: Tabela de atributos do jogos

Nome	**Tipo**	**Descrição**
start	Booleano (*true/false*)	Indica se o jogo está começando. "*true*"indica que uma nova sessão foi iniciada.
var_name	String	Nome da variável a ser controlada, variável "*Score*"usada nos testes.
new_value	Inteiro	Atualização do valor de "*Score*".
set_point	Inteiro	Valor desejado para "*Score*".
player	String	Nome do jogador.
action	String (*Tipo_A, Tipo_B, Tipo_C*)	Tipo de incêndio combatido pelo jogador.
result	Inteiro (*0 ou 1*)	Resultado da ação, onde "*1*"indica sucesso e "*0*"fracasso.
moveSpeed	Float	Velocidade atual do jogo.
date	Datetime	Data e hora atual da sessão.
lost_	Inteiro	Cliques de mouse desperdiçados.
balanceUsed	Booleano (*true/false*)	Tipo de balanceamento usado, "*true*"para PID e "*false*"para balanceamento tradicional.

Fonte: Do Autor.

5.1.2 Execução

Na Figura 5.2, estão apresentadas as instruções do jogo onde o incêndio só poderá ser combatido na zona verde. Um incêndio Tipo_A (sólido combustível) está circulado em amarelo. No HUD (*Heads-Up Display* / tela de informações) estão os "Acertos"(incêndios combatidos corretamente), "Erros"(incêndios combatidos incorretamente) e "*Score*"(diferença entre acertos e erros). Na parte superior central, está o *timer* regressivo do jogo. O *Score* é a variável que é enviada em tempo real ao servidor para balanceamento. O valor desejado para a pontuação é de "6"pontos. Caso o *Score* esteja abaixo de 6, a velocidade do jogo e a velocidade de *spawn* de objetos serão reduzidas. Caso esteja maior que "6", as velocidades aumentarão.

Figura 5.2: Proposta do Jogo

Fonte: Do Autor.

Para o balanceamento PID foram utilizados os seguintes parâmetros:"**kp**": 1.0, "**ki**": 0.05, "**kd**": 0.8 e "**setpoint**": 6.0.
O experimento de *Analytics* foi configurado para avaliar continuamente o perfil do jogador e ajustar a dificuldade do jogo por meio de um algoritmo PID. O servidor, equipado com funcionalidades de análise e balanceamento das variáveis enviadas, ajusta as condições do jogo em tempo real.

Mecânica de Coleta de Dados

Os principais **Dados de Desempenho**: Pontuação e Velocidade do mapa.
Com duração de 5 minutos na rodada, onde foram coletados dados comparativos entre o desempenho tradicional e o desempenho sob influência do controlador PID. Para

o autobalanceamento tradicional, também foram coletados *Score* (pontuação) para o balanceamento. O ajuste de balanceamento foi de ± 0.5 na velocidade do jogo e no tempo de *spawn* de objetos.

5.1.3 Resultados estatísticos

Como primeira análise, foi determinada a normalidade das funções Pontuação x Velocidade. Esta análise será utilizada para determinar quais tipos de análise de correlação serão utilizados. A análise dos modos Linear e PID na tabela 5.4 revelou que, em ambos os casos, as variáveis X (pontuação) e Y (velocidade) não seguem uma distribuição normal, conforme indicado pelos baixos p-valores dos testes de normalidade. No modo Linear, a correlação de Spearman apresentou um valor de 0,96, enquanto no modo PID, o valor foi de 0,93. Esses resultados sugerem uma forte correlação monotônica em ambos os modos, com o modo Linear exibindo uma correlação ligeiramente mais alta.

Tabela 5.4: Análise de Normalidade e Correlação para Modos Linear e PID

Modo	**Análise de Normalidade**	**Correlação Adequada (Valor)**
Linear	- X: Pontuação não é normal (p-valor: 6.506e-10) - Y: Velocidade não é normal (p-valor: 1.947e-10)	Spearman (0.96)
PID	- X: Não é normal (p-valor: 4.836e-03) - Y: Não é normal (p-valor: 2.243e-03)	Spearman (0.93)

Fonte: Do Autor.

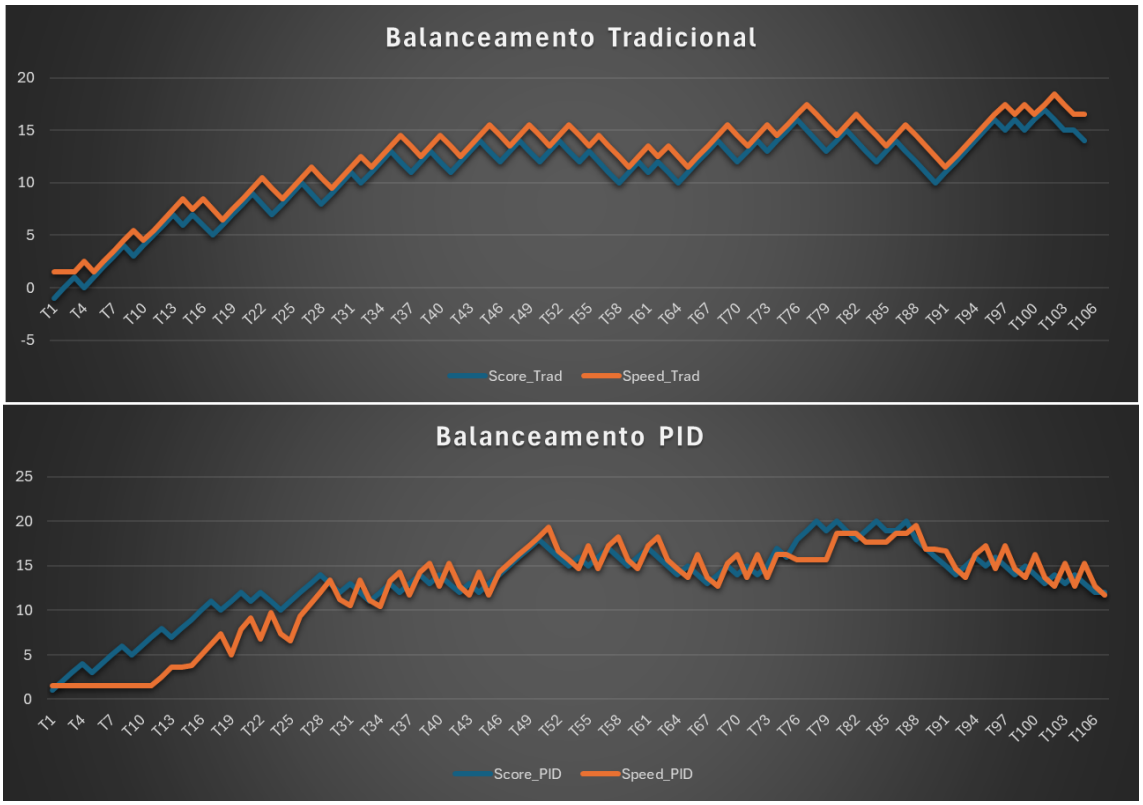
5.1.4 Análise de Dados e Resultados

Comparação entre os métodos Tradicional x PID
A Figura (5.3) apresenta uma comparação entre os métodos Linear e PID, o primeiro gráfico exibe o comportamento da velocidade (Eixo y) e pontuação (Eixo x), o gráfico mais abaixo exibe o comportamento do PID para as mesmas variáveis.

1. **Método Tradicional**

   No balanceamento Tradicional, o gráfico de Speed_Trad mostrou uma correlação direta (Tabela 5.4) com Score_Trad. Isso indica que as alterações na velocidade do jogo são diretamente proporcionais às alterações na pontuação do jogador. A abordagem tradicional segue uma reação linear, onde aumentos ou diminuições na pontuação resultam em ajustes proporcionais na velocidade do jogo, mantendo uma resposta consistente e previsível.

Figura 5.3: PID x Tradicional

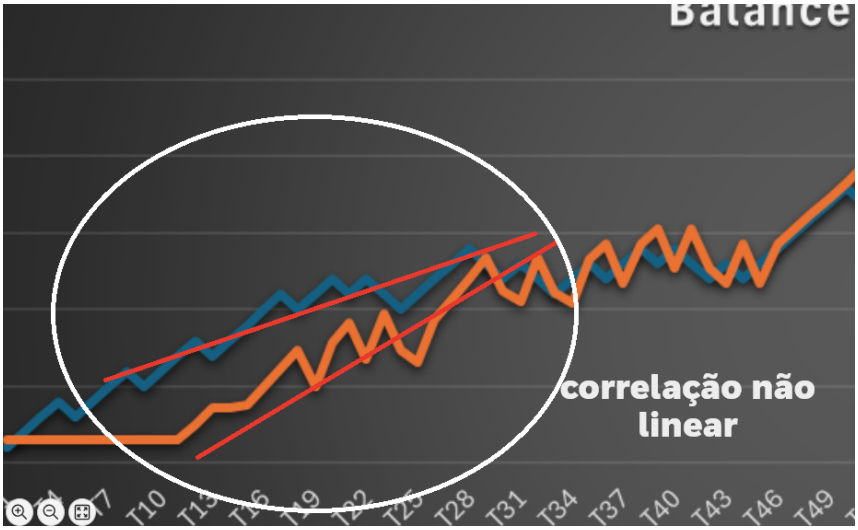


Fonte: Do Autor.

2. **Método PID**

   Por outro lado, por causa dos componentes Integrativos e Derivativos, o método PID apresentou uma resposta mais complexa e dinâmica. Apesar da forte correlação (Tabela 5.4), visualmente, pode se observar ajuste pontuais mais agressivos às mudanças em Score_PID (Figura 5.4), sugerindo um ajuste antecipatório na velocidade. Esta abordagem pode ser mais eficaz em cenários onde a adaptação rápida seja importante. A seguir, serão feitas análises matemáticas de correlação em diferentes faixas de velocidade.

Figura 5.4: PID: *Score x Speed*



Fonte: Do Autor.

A Tabela 5.5 oferece a variação de correlação em diferentes faixas de velocidade. Faixa de pontuação total de 1.5 até 20 (faixa da menor pontuação para maior). A partir da velocidade 6 (acima do *setpoint*), observa-se uma redução gradual da correlação até o resultado de 0.78.

Tabela 5.5: Correlação (Spearman) para diferentes pontuações - PID

Faixa de Pontuação	**Correlação de Spearman (Valor)**
1.5 até 20	0.94
6 (setpoint) até 20	0.87
11 até 20	0.78

Fonte: Do Autor.

A comparação entre os dois métodos revela diferenças significativas na abordagem e eficácia do balanceamento. Enquanto o método Tradicional oferece uma abordagem mais estável e previsível, o método PID pode proporcionar uma experiência de treinamento mais desafiadora e adaptativa, que ajusta a dificuldade de maneira mais eficiente e em tempo real, em resposta às ações do jogador.

O experimento desenvolvido para avaliar o sistema de treinamento baseado em jogos apresentou algumas limitações importantes durante a fase inicial do projeto e que requerem atenção para futuras iterações e expansão do projeto.

Uma das principais limitações encontradas relaciona-se ao uso de software e ferramentas específicas. A escolha das tecnologias foi guiada principalmente pela facilidade e acessibilidade das plataformas, o que, embora benéfico para agilizar o desenvolvimento

inicial, pode não ser ideal para a escalabilidade ou integração futura em ambientes de produção mais complexos. Essa dependência pode restringir a capacidade do sistema de se adaptar a novas tecnologias ou de ser integrado com sistemas existentes em ambientes corporativos.
O estudo foi conduzido em uma escala reduzida. Esta abordagem limitada impede a validação completa do experimento proposto em cenários mais diversificados e com múltiplos usuários.
A simplicidade do jogo usado nessa etapa também pode representar uma limitação significativa. Embora jogos simples possam ser úteis para focar em aspectos específicos do treinamento, eles podem não capturar a complexidade e a variedade de desafios encontrados em ambientes de trabalho reais.

5.2 Estudo Experimental – SpacePilot

O jogo *SpacePilot* foi projetado como um *endless run* de entretenimento, onde o jogador controla uma nave espacial que deve desviar de obstáculos para maximizar sua pontuação. Esta seção apresenta as etapas para implementação do método de balanceamento, incluindo a abordagem estática (fácil), linear e o controlador PID.

5.2.1 Definição e Planejamento do Estudo

A definição do estudo apresenta os objetivos gerais e específicos da pesquisa.O objetivo (*goal*) pode ser descrito como: *avaliar o comportamento de diferentes métodos de balanceamento de dificuldade (Fácil, Linear e PID) no jogo SpacePilot*; com o propósito de *comparar o impacto de cada método no desempenho e na experiência do jogador em função de métricas de desempenho*; no que diz respeito a *(Q1) correlação entre velocidade e pontuação, (Q2) análise de estabilidade no ajuste de dificuldade e (Q3) possíveis implicações de uso em diferentes perfis de jogadores*; do ponto de vista de *pesquisadores, desenvolvedores de jogos e entusiastas de jogos digitais*, no contexto de um *endless run de entretenimento focado em coleta de dados para análise de balanceamento*.
De acordo com a *estrutura de avaliação* (inspirada em Basili (Basili *et al.*, 1992)), as questões de pesquisa (*question*) que se pretende responder são as seguintes:

- **Q1**: Quais são as correlações entre a *Velocidade* e a *Pontuação* quando se utiliza o balanceamento fácil (estático), linear e o PID?
- **Q2**: Em que medida cada método (linear e PID) gera maior razão entre acertos e erros?

Para cada uma dessas perguntas, foram pensados indicadores (*metrics*) que auxiliarão na mensuração dos resultados obtidos.

Quando se investiga a *correlação entre Velocidade e Pontuação* (Q1), busca-se avaliar se os valores de *score* e *velocidade* estão positivamente relacionados, em que nível e em qual método de balanceamento (Linear ou PID). Dessa forma, a métrica **(M1)** utilizada é o *coeficiente de correlação* (Pearson, Spearman ou Kendall).

No caso de *acurácia* (Q2), pretende-se observar variações na pontuação do jogo para ajustes na velocidade. Assim, do ponto de vista quantitativo, pode-se adotar uma métrica semelhante, por exemplo, **(M2)** = *número de acertos* / *número de erros*, que indica se o ajuste de dificuldade contribuiu para o desempenho do jogador.

Para (M1) e (M2), considera-se que a quantidade de *instâncias* (ou rodadas de jogo) esperadas seja igual ao número de sessões realizadas pelos participantes, cada uma com aproximadamente 2,5 minutos de duração. Essa definição baseia-se na estrutura do *SpacePilot*, no qual cada sessão gera, via requisições `GET`, um registro contendo: *Pontuação*, *Velocidade*, *Tempo Restante* e *Método de Balanceamento*. Justifica-se por se tratar de dados contínuos, coletados ao longo de cada partida. Estatisticamente, tais métricas podem ser tratadas na escala de *intervalo* (ou *razão*, dependendo da variável), permitindo análises *descritivas*, *inferenciais* e *correlacionais* (Shapiro-Wilk, Spearman etc.).

Além das métricas quantitativas, serão realizadas análises qualitativas por meio de *questionários* , avaliando a percepção dos jogadores quanto à dificuldade e fluidez do jogo em ambos os métodos (Fácil, Linear e PID). Entende-se que uma abordagem mista, envolvendo dados *quali-quanti*, poderá oferecer uma compreensão mais abrangente dos resultados, enriquecendo a interpretação dos achados e destacando *benefícios, limitações e percepções* dos participantes (Recker, 2013).

Planejamento do Estudo

Para conduzir a avaliação proposta, definiu-se um planejamento de estudo que estabelece em detalhes o *contexto, procedimentos e critérios de seleção de participantes*, bem como a configuração do jogo *SpacePilot*. Dessa forma, pretende-se garantir a clareza sobre como a pesquisa foi conduzida, possibilitando a replicação dos procedimentos por pares científicos e a análise comparativa entre o balanceamento Fácil, Linear e o PID.

Os participantes foram selecionados por *conveniência*, levando em consideração a disponibilidade de jogadores de diferentes perfis (iniciantes / experientes). Eles foram convidados a participar voluntariamente do estudo após *realizarem uma breve sessão de instrução* sobre o funcionamento básico do *SpacePilot*, de forma que todos entendessem o objetivo do jogo (*endless run* para desviar de obstáculos) e as diferentes abordagens de ajuste de dificuldade. A escolha desse perfil heterogêneo de jogadores justifica-se

para avaliar se há diferenças de percepção e desempenho em relação a cada método de balanceamento.

Para analisar *(Q1) correlação* e *(Q2) acurácia* em diferentes contextos, optou-se por distribuir **três métodos de balanceamento** (*Fácil*, *Linear* e *PID*) entre os participantes. Por exemplo, parte deles (Grupo A) jogou apenas com o *método Fácil*, enquanto outra parte (Grupo B) jogou exclusivamente com o *método Linear* e a outra parte (Grupo C) jogou exclusivamente com o *método PID*. Essa divisão permite investigar se as dinâmicas de cada método são aplicáveis a perfis distintos de jogadores e quais fatores podem influenciar os resultados (como curva de aprendizagem e complexidade do controlador).

O estudo foi projetado para ocorrer em *3* fases principais, conforme a Tabela 5.6.

Tabela 5.6: Etapas de Execução do Estudo *SpacePilot*

Etapa	Duração (min)	Descrição
E1	<5>	*Treinamento*, instruções de como desviar de obstáculos
E2	<10>	*Aplicação prática*: execução do jogo *SpacePilot*
E3	<5>	*Questionário* : percepções, usabilidade, sugestões

Fonte: Do Autor.

Na primeira etapa (E1), houve o treinamento e a familiarização com os controles do jogo, bem como a explicação sobre cada método de balanceamento. Já na segunda etapa (E2), ocorreu a execução do jogo propriamente dito, registrando em planilhas as variáveis de desempenho. Por fim, a terceira etapa (E3) compreendeu a aplicação de questionários , visando coletar dados sobre as percepções e preferências dos participantes.

Para apoiar a execução de cada etapa, foram disponibilizados aos participantes *tutorial* e um breve *guia* dentro do próprio jogo *SpacePilot*. Na etapa E2, utilizou-se **requisições GET** para **capturar dados de Pontuação, Velocidade, Tempo Restante e Método** em intervalos regulares, sendo tudo automaticamente registrado em uma planilha Google. Esses registros permitiram observar o comportamento em tempo real de cada participante, bem como embasar as análises das métricas (M1, M2).

Na etapa E3, aplicou-se um questionário composto por *6* questões, contemplando tanto *questões objetivas* quanto *questões subjetivas* (relatos sobre fluidez e engajamento). As respostas foram coletadas por meio de um *formulário online GoogleForms*, de forma que fosse possível consolidar e comparar os dados.

Antes da condução oficial do estudo, foi realizado um *pré-teste* com 2 participantes para validar o planejamento, identificando possíveis lacunas na coleta de dados e pontos de melhoria nas instruções do *SpacePilot*. As sugestões e correções feitas pelos participantes do piloto foram incorporadas ao planejamento final, de modo que *possíveis erros de registro* ou *dificuldades na interface* fossem minimizados.

A coleta de dados ocorreu de duas maneiras principais: (1) **planilha** com as informações geradas pelas requisições `GET` (pontuação, velocidade, método, tempo restante) e (2) **respostas do questionário** (formulário online) . Para as análises quantitativas, utilizou-se *Python* para o teste de normalidade Shapiro-Wilk para definir a correlação (Pearson, Spearman ou Kendall). Para as análises qualitativas, optou-se por **análise de conteúdo**, categorizando os principais relatos sobre a experiência de jogo. Definiu-se um nível de significância de $\alpha = 0,05$ para testes de hipóteses estatísticas em relação às diferenças entre Linear e PID.

Por fim, foram consideradas as possíveis ameaças à validade do estudo, tais como *ameaças de conclusão, internas, de construção e externas* (Creswell; Creswell, 2017). Na Tabela 5.7, constam as principais ameaças identificadas, bem como as estratégias adotadas para mitigá-las. Por exemplo, conduzir o *pré-teste* com participantes representativos, revisar a instrumentação de coleta (script do `GET` e planilhas Google) e planejar a diversidade do perfil de jogadores.

Tabela 5.7: Ameaças à Validade e Tratamentos

Tipo	**Ameaça**	**Descrição**	**Tratamento**
Conclusão	Poder estatístico	Amostra limitada pode afetar conclusões	Estimação mínima de participantes e adoção de métodos não paramétricos (Spearman)
Interna	Falta de treinamento	Participantes podem não compreender a mecânica de balanceamento	Pré-teste, tutorial e guia dentro do jogo
Construção	Instrumentação	Falhas no envio de dados via requisição `GET` e planilhas	Verificação manual periódica, logs automáticos e revisão do código
Externa	Generalização	Foco em apenas um tipo de jogo *(SpacePilot)*	Replicações futuras em jogos do mesmo gênero (endless run)

Fonte: Do Autor.

5.2.2 Execução

Mecânica de Jogo e Balanceamento

A experiência de jogo em *SpacePilot* utiliza dois métodos principais de balanceamento para ajustar a dificuldade com base no desempenho do jogador descritos na Tabela 5.8:

Tabela 5.8: Balanceamento da velocidade

Método	**Descrição**
Linear	A velocidade da nave aumenta proporcionalmente à pontuação do jogador, definida pela equação: $$\text{Velocidade} = \frac{1}{6} \times \text{Pontuação}$$ O valor $1/6$ foi utilizado como ponto de partida, representando a relação entre a menor (1) e a maior velocidade (6).
PID	Um controlador PID ajusta dinamicamente a velocidade da nave com base na pontuação do jogador, utilizando os parâmetros: • $K_p = 0.03$: Proporcional • $K_i = 0.001$: Integral • $K_d = 0.03$: Derivativo • Setpoint inicial = 20.0

Fonte: Do Autor.

***Setpoint* variável**

O *setpoint* para o ajuste PID aumenta ou diminui em 20 pontos a cada nível alcançado, ou perdido, promovendo ajustes dinâmicos na dificuldade. Esse valor foi definido em decorrência do estudo desenvolvido no jogo InfintyFire.

5.2.3 Coleta de Dados e Integração

O estudo piloto foi realizado entre os meses de outubro de 2024 e dezembro de 2024, seguindo as mesmas etapas previstas para o estudo quase-experimental. Contou-se com a participação de 163 pessoas, de diferentes áreas de atuação e diferentes níveis de experiência e jogos digitais, que foram convidadas a responder a um questionário inicial. Todos os participantes receberam instruções gerais sobre a pesquisa e sobre a forma de execução do jogo por meio de um questionário online.

Para verificar o efeito das diferentes configurações de balanceamento no jogo, os participantes foram divididos em grupos de acordo com o tipo de método de

balanceamento empregado (Fácil, Linear ou PID), de modo a comparar o desempenho em diferentes cenários. Essa divisão permitiu isolar as variáveis referentes ao balanceamento e analisar as respostas de forma segmentada, possibilitando uma avaliação mais precisa de como cada método impacta o comportamento dos jogadores.

Após responderem ao questionário inicial, os participantes iniciaram a execução do jogo, seguindo as instruções fornecidas no próprio formulário e pela instrução interna do jogo. Após o fim da partida, concluíram o preenchimento do questionário, registrando suas impressões e gerando os dados necessários para a análise. Durante a execução do jogo dados como pontuação total acumulada (Pontuação), velocidade ajustada pelo balanceamento (Velocidade), valor do *timer* regressivo (Tempo Restante) e método de balanceamento empregado (Método de Balanceamento) foram coletados em tempo real por meio de requisições do tipo `GET` e enviados, de forma periódica, para uma planilha do Google. Esse procedimento facilitou o monitoramento contínuo do desempenho dos participantes e possibilitou análises detalhadas e prosseguir com as etapas seguintes do experimento.

5.2.4 Análise de Dados e Resultados

Após a execução do experimento, os dados coletados por meio da **planilha Google** (enviados via requisições `GET` e complementados pelos *logs* do jogo e respostas do questionário) puderam ser computados e analisados por estratégias de análise quantitativa, enquanto as percepções e observações apontadas nas *questões subjetivas* do formulário foram tratadas qualitativamente. Todos os dados foram agrupados e, para as análises quantitativas, foi utilizado o **Python** (6.5). Para a análise qualitativa, adotou-se a **análise de conteúdo**, categorizando as principais opiniões e comentários dos participantes sobre o jogo e o balanceamento. Os dados foram sumarizados em tabelas e gráficos para prover uma interpretação simplificada. Todas as análises foram extraídas exclusivamente a partir das respostas dos *participantes* que fizeram parte do estudo.

Como primeira análise, foi determinada a normalidade das funções Pontuação x Velocidade. Esta análise será utilizada para determinar quais tipos de análise de correlação serão utilizados. Na Tabela (5.9) estão os resultados dos testes Shapiro-Wilk.

Q1 – Correlação entre Velocidade e Pontuação.

A primeira análise realizada consistiu em verificar a relação entre a velocidade do jogo e a pontuação obtida pelos participantes (*Q1*). Como definido no delineamento, a métrica de correlação (Pearson, Spearman ou Kendall) foi escolhida com base na *normalidade* dos dados, verificada pelo **teste de Shapiro-Wilk**. Ao não se confirmar a normalidade (p-value $< 0{,}05$), aplicou-se a correlação não-paramétrica de **Spearman** para avaliar a força e direção da associação entre *Velocidade* e *Pontuação* nos modos de balanceamento

Tabela 5.9: Resultados do Teste de Shapiro-Wilk de Normalidade

Modo	Variável	Resultado	p-valor
Linear	X (Pontuação)	Não é normal	2.579×10^{-12}
	Y (Velocidade)	Não é normal	1.772×10^{-9}
PID	X (Pontuação)	Não é normal	5.400×10^{-6}
	Y (Velocidade)	Não é normal	1.348×10^{-15}
Fácil	X (Pontuação)	Não é normal	2.884×10^{-13}
	Y (Velocidade)	NA [1]	N/A

Fonte: Do Autor.

Linear e *PID*.

5.2.4.0.1 Q2 – Acurácia (Erros vs. Acertos).

A segunda análise (Q2) foi realizada para verificar a **acurácia** dos participantes em relação aos obstáculos do jogo, ou seja, a proporção de acertos comparada aos erros em cada modo de balanceamento. A Tabela 5.10 apresenta o percentual médio de *acertos* para cada método (*Fácil, Linear, PID*).

Tabela 5.10: Porcentagem de Acertos em Cada Modo de Balanceamento (Q2)

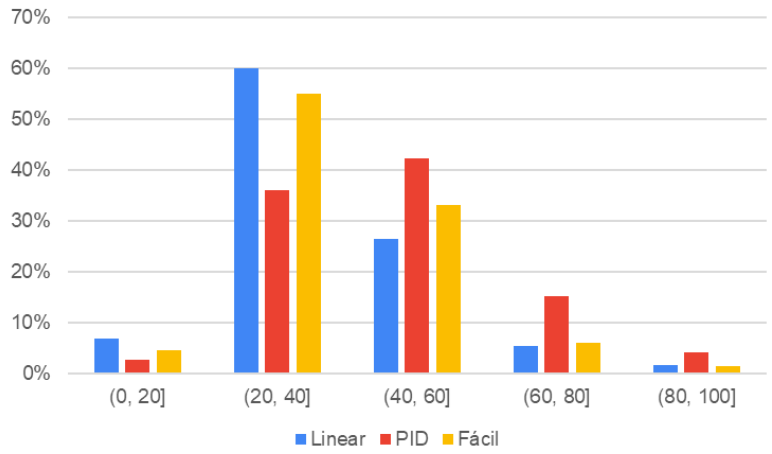
Modo de Balanceamento	% de Acertos (média)
Fácil	61%
Linear	66%
PID	68%

Fonte: Do Autor.

5.2.4.1 Frequência de pontuação por tipo de balanceamento

Essa visualização permite verificar qual valor de pontuação foi predominante em cada tipo de balanceamento. Para essa verificação, as pontuações foram divididas em faixas (de 0 até 20 pontos / de 20 até 40 pontos / de 40 até 60 pontos / de 60 até 80 pontos / de 80 até 100 pontos). Abaixo são apresentados os gráficos indicando as frequências das pontuações em cada tipo de autobalanceamento, Modo Fácil, Modo Linear e Modo PID . A Tabela (5.11) apresenta de forma detalhada a frequência das faixas de pontuação.
Conforme apresentado na Figura 5.5, a distribuição de pontuação foi agrupada em faixas de 20 pontos para cada um dos três modos de autobalanceamento (*Fácil*, *Linear* e *PID*). Observa-se que o modo **Fácil** concentra a maior parcela de jogadores (55%) na faixa de 20 a 40 pontos, com poucos participantes alcançando pontuações entre 80 e 100 pontos (1%). Isso reflete a **velocidade fixa** adotada pelo modo Fácil, limitando, em geral, a

Figura 5.5: Comparação: *Pontuação X Ocorrências*



Fonte: Do Autor

Tabela 5.11: Frequência de Pontuação: Fácil, Linear e PID

Faixa de Pontuação	**Modo Fácil (%)**	**Modo Linear (%)**	**Modo PID (%)**
0 – 20 pontos	5	7	3
20 – 40 pontos	55	60	36
40 – 60 pontos	33	26	42
60 – 80 pontos	6	5	15
80 – 100 pontos	1	1	4

Fonte: Do Autor.

maximização do *score*.

No modo **Linear**, há uma tendência similar, mas com leve deslocamento para a faixa de 20 a 40 pontos (60%). Em relação ao modo **Fácil**, nota-se maior dispersão dos jogadores nas faixas de *40 a 60* (26%) e *60 a 80* (5%), sugerindo que, embora o balanceamento linear possibilite ganhos de pontuação mais altos, ainda há predomínio na faixa intermediária.

Por outro lado, no modo **PID** observa-se uma maior concentração de jogadores na faixa entre *40 e 60 pontos* (42%), ultrapassando o número de jogadores na faixa de *20 a 40* (36%). Além disso, 15% dos jogadores atingem a faixa *60 a 80*, e 4% a faixa de *80 a 100*. Esse resultado indica que o ajuste dinâmico proporcionado pelos componentes proporcional, integral e derivativo (*Kp, Ki, Kd*) favorece uma melhor adaptação às habilidades dos participantes, possibilitando, em média, pontuações mais elevadas.

5.2.4.2 Medida de Correlação entre Velocidade x Pontuação

As técnicas matemáticas empregadas para análise de correlação no código incluem métodos estatísticos amplamente utilizados na avaliação de relações entre variáveis. O teste de normalidade Shapiro-Wilk (Shapiro; Wilk, 1965) é aplicado inicialmente para verificar se os dados seguem uma distribuição normal, critério importante na escolha do

método de correlação mais apropriado. Para mensurar a força e a direção da associação entre variáveis, podem ser utilizados coeficientes de correlação: Pearson (Pearson, 1895), que mede relações lineares em dados normalmente distribuídos; Spearman (Spearman, 1904), que avalia relações monotônicas [2], sendo robusto para dados não lineares ou não normais; e Kendall Tau (Kendall, 1938), ideal para amostras pequenas ou dados categóricos ordenados. Essas abordagens oferecem uma visão abrangente das dependências entre as variáveis e permitem a seleção do método mais adequado com base na natureza dos dados analisados.

Neste estudo, utilizou-se um código python para o cálculo de correlação, código descrito no Apêndice A 6.5. Para analisar a relação entre a velocidade e a pontuação nos casos de autobalanceamento Linear e PID. Para todos os casos, o cálculo de Shapiro-Wilk não obteve normalidade (5.9), portanto, foi utilizada a correlação de Spearman.

O dados de velocidade e pontuação foram agrupados por tipo de balanceamento, todas as ocorrências foram calculadas seguindo a série temporal de registro na planilha.

A correlação entre *Velocidade* e *Pontuação* no **modo Linear** alcançou um coeficiente de **0,96**, demonstrando **forte relação positiva** entre a velocidade e o *score*. Ou seja, quanto maior a pontuação configurada no modo Linear, maior tende a ser a velocidade do jogo.

Já o **modo PID** apresentou um coeficiente de **0,84**, também caracterizando uma **correlação positiva** entre velocidade e pontuação, porém menos intensa do que no modo Linear. Este resultado condiz com a própria natureza do controlador PID, cujo ajuste envolve termos proporcional, integral e derivativo.

No **modo Fácil**, como o comportamento da função é constante (para qualquer pontuação, a velocidade será sempre de 1), não foi possível calcular a correlação.

A Tabela 5.12 apresenta de forma resumida os coeficientes de correlação (ρ) identificados para todas as ocorrências em cada tipo de autobalanceamento.

Tabela 5.12: Análise de Correlação (Q1) entre Velocidade e Pontuação

Modo de Balanceamento	**Coef. Spearman**	**Interpretação**
Fácil	-	Não aplicável (velocidade fixa)
Linear	0,96	Correlação positiva forte
PID	0,84	Correlação positiva moderada-forte

Fonte: Do Autor.

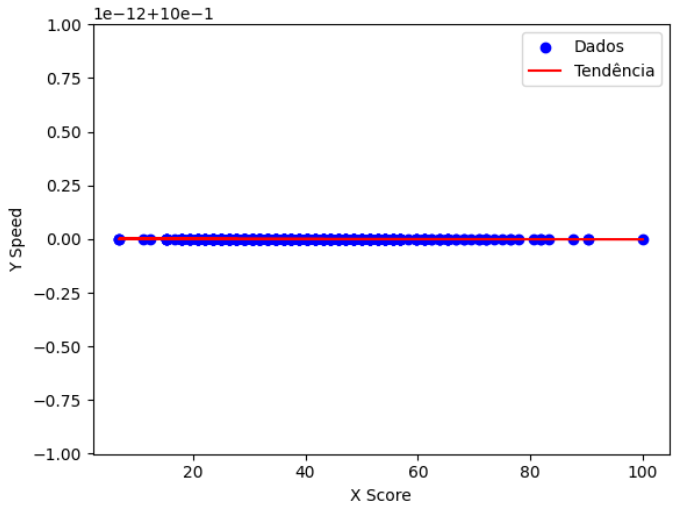
Abaixo serão apresentados os gráficos do tipo *ScatterPlot* (ou Gráficos de Dispersão), que são representações visuais usadas para exibir a relação entre duas variáveis quantitativas em um plano cartesiano. Cada ponto no gráfico representa uma observação, onde a

[2]Relações monotônicas são aquelas em que as variáveis mantêm uma direção consistente sempre crescente ou sempre decrescente , sem necessariamente apresentar uma variação proporcional.

posição horizontal (x) corresponde ao valor de uma variável e a posição vertical (y) ao valor da outra variável.

Na Figura 5.6, é apresentado um gráfico de dispersão (*ScatterPlot*) que ilustra a relação entre *Velocidade* (eixo Y) e *Pontuação* (eixo X) no modo *Fácil*. Esse gráfico permite observar o comportamento constante da velocidade em relação à pontuação, ou seja, a velocidade permanece sempre no mesmo valor, independentemente de quanto o jogador pontue. Os pontos azuis no gráfico representam os dados coletados durante o experimento, enquanto uma linha vermelha foi adicionada para indicar a tendência geral desses dados, facilitando a compreensão visual do padrão fixo.

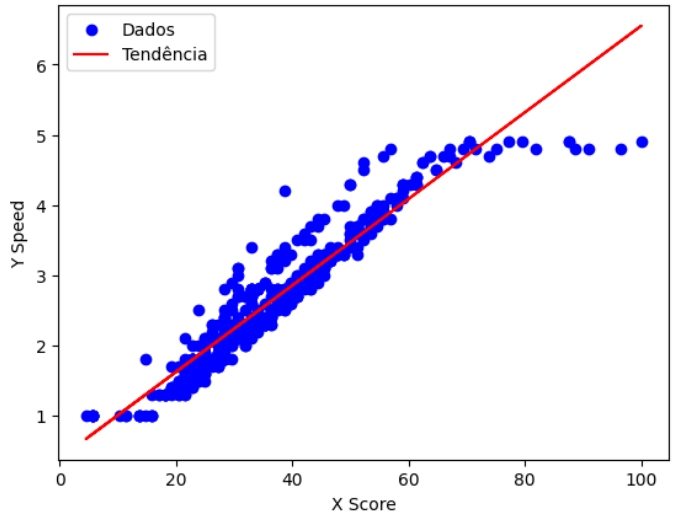
Figura 5.6: ScatterPlot - Modo Fácil



Fonte: Do Autor.

Na Figura 5.7, seguindo o mesmo padrão de dados, está representado o gráfico ScatterPlot no modo Linear, observa-se um comportamento gradual e crescente na relação Velocidade (eixo x) x Pontuação (eixo y).

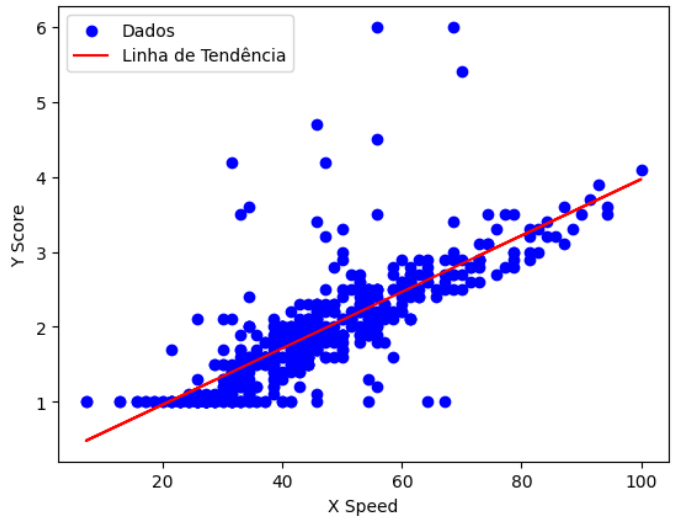
Figura 5.7: ScatterPlot - Modo Linear



Fonte: Do Autor.

Na Figura 5.8, seguindo o mesmo padrão de dados, está representado o gráfico ScatterPlot no modo PID, observa-se um comportamento gradual e crescente, mas com pontos de ajuste fora do padrão de dispersão na relação Velocidade (eixo x) x Pontuação (eixo y).

Figura 5.8: ScatterPlot - Modo PID



Fonte: Do Autor.

Correlação em diferentes faixas de pontuação no método PID

Como observado no jogo InfinityFire, a correlação reduz quando os valores de pontuação estão acima do *setpoint*, a Tabela 5.13 estratifica a variação da correlação para faixa de valores acima dos *setpoints*. A partir dos dados do estudo com o InfintyFire, foi implementado no jogo SpacePilot diferentes níveis de *setpoints*. A partir de 20 pontos, a

cada 20 pontos o *setpoint* é atualizado (20, 40, 60 e etc). Para verificação da correlação fora dos parâmetros de *setpoint*, foi analisada a correlação em patamares de pontuação com 5 pontos acima do *setpoint* (25, 45, 65 e 85). O objetivo e verificar a correlação ou o comportamento dos ajustes feitos pelo PID fora do *setpoint* determinado.

Tabela 5.13: Correlação entre Velocidade e Pontuação para Diferentes Faixas

Faixa de Pontuação	**Spearman(Velocidade x Pontuação)**
20 a 100	0.83
25 a 40	0.53
45 a 60	0.24
65 a 80	0.50
85 a 100	NA - baixa amostragem.

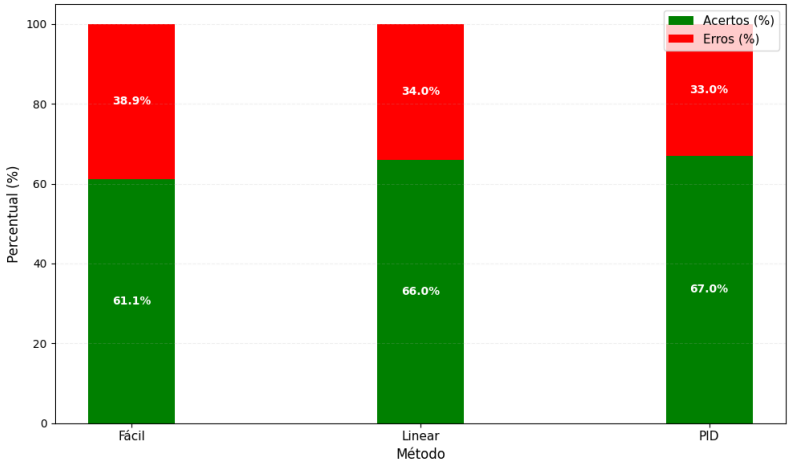
Fonte: Do Autor.

5.2.4.3 Erros x Acertos

A análise de proporção entre duas variáveis é uma técnica estatística utilizada para avaliar a relação quantitativa entre os valores de duas variáveis de interesse. No contexto de desempenho em sistemas de autobalanceamento, essa análise consiste em calcular a razão entre o número de acertos e erros registrados em diferentes condições experimentais ou modos operacionais, permitindo identificar padrões de eficiência ou precisão.

A análise da proporção entre acertos e erros nos diferentes modos de autobalanceamento revelou diferenças significativas. No modo **Fácil**, a proporção de acertos por erros foi de aproximadamente **1,57**, indicando uma leve predominância de acertos em relação aos erros. Já no modo **Linear**, essa proporção aumentou para **1,94**, sugerindo uma maior precisão por parte dos jogadores. Por fim, no modo **PID**, a proporção alcançou **2,02**. Para melhor visualização, a Figura 5.9 apresenta o Gráfico de Barras Empilhadas com os resultados.

Figura 5.9: Proporção Acertos x Erros



Fonte: Do Autor.

5.2.4.3.1 Estatística inferencial - Comparativo Linear x PID

São apresentadas as estatísticas descritivas e inferenciais dos dados coletados para os dois modos de autobalanceamento (*Linear* e *PID*).
Após a verificação de normalidade (Tabela 5.9) as seguintes técnicas estatísticas foram aplicadas:

- **Cohens d**: Métrica de tamanho de efeito que quantifica a magnitude da diferença entre duas médias (Cohen, 1988).
- **Mann-Whitney U (Wilcoxon rank-sum)**: Teste não-paramétrico para comparar se duas amostras provêm da mesma distribuição, adequado quando não se pode assumir normalidade (Mann; Whitney, 1947).

A Tabela 5.14 apresenta as estatísticas descritivas (média e desvio-padrão) para pontuação nos modos *Linear* e *PID*. Em seguida, mostra-se o valor de Cohens d e os resultados de Mann-Whitney U para pontuação e velocidade.
Observa-se que, embora a pontuação média no modo *PID* (46.6) seja numericamente maior que no modo *Linear* (36.7), o teste Mann-Whitney U indica que **não é possível afirmar, com 95% de confiança, que há uma diferença estatisticamente significativa** entre as pontuações médias dos dois modos ($p \approx 0.2385$). Já para a *velocidade*, o resultado ($p < 0.05$) sugere que **é possível afirmar, com ao menos 95% de confiança, que existe uma diferença estatisticamente significativa** entre os dois modos nesse aspecto.
Erros e Acertos
Os achados quantitativos sugerem que, sob o ponto de vista de acertos/erros, ambos os modos apresentam desempenho semelhante.

Tabela 5.14: Estatísticas Descritivas e Resultados Infenciais para Pontuação e Velocidade

Métrica	**Linear**	**PID**
Média (Pontuação)	36.7	46.6
Desvio-Padrão (Pontuação)	14.77	16.42
Cohen's d (Pontuação PID vs. Linear) = 0.640 **Cohen's d (Velocidade PID vs. Linear)** = -0.772		
Mann-Whitney U (Pontuação): $U = 110011.5$, $p = 0.2385$ Não há diferença significativa para a pontuação ($p > 0.05$).		
Mann-Whitney U (Velocidade): $p = 0.0000$ Há diferença significativa para velocidade ($p < 0.05$).		

Fonte: Do autor.

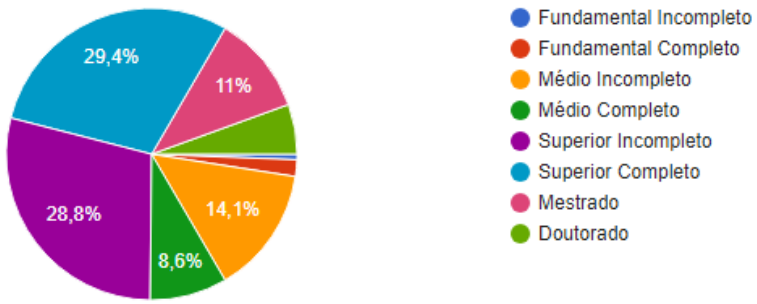
5.2.5 Resultados Qualitativos

Para a análise qualitativa, foram considerados os perfis dos participantes, suas preferências de jogo e percepção sobre os modos de autobalanceamento. As informações detalhadas são apresentadas nas subseções e figuras a seguir, de forma a oferecer uma visão abrangente sobre os resultados.

Perfil dos Participantes

O perfil dos participantes foi analisado com base em critérios como grau de formação, localização geográfica e gênero. A Figura 5.10 apresenta a distribuição do grau de formação dos participantes. Observa-se uma ampla variação entre Ensino Médio, Ensino Superior e Pós-Graduação.

Figura 5.10: Grau de formação dos participantes.



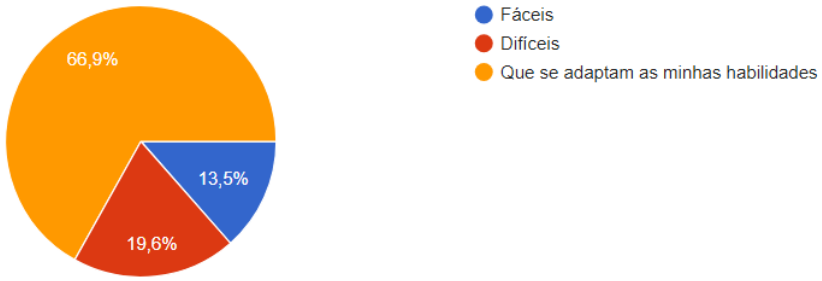
Fonte: Do Autor.

A distribuição geográfica revela que 65,6% dos respondentes residem no estado do Rio de Janeiro, 14,1% em Minas Gerais e 7,4% em São Paulo. Em relação ao gênero, 61% dos participantes se identificam como masculino, 38% como feminino, e 1% como não binário ou não declarado.

Preferências por Modo de Jogo

Os participantes foram questionados sobre suas preferências em relação aos modos de jogo. A Figura 5.11 ilustra que 67% preferem "jogos que se adaptam às minhas habilidades", enquanto 13% optam por "jogos fáceis"e 20% preferem "jogos difíceis".

Figura 5.11: Preferência por modos de jogo.

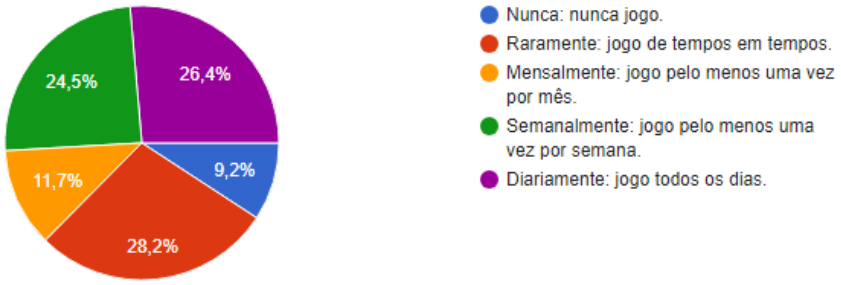


Fonte: Do Autor.

Frequência com que Joga

A frequência com que os participantes jogam é apresentada na Figura 5.12. Observa-se que a maioria joga regularmente em intervalos semanais, mas há uma variação considerável entre os diferentes perfis de engajamento.

Figura 5.12: Frequência com que os participantes jogam.



Fonte: Do Autor.

Percepção sobre os Modos de Autobalanceamento

A Tabela 5.15 apresenta as percepções sobre os modos de autobalanceamento.
Observa-se que o modo **PID** foi considerado apropriado por 68% dos respondentes, enquanto o modo Linear teve 64% de aprovação. O modo Fácil, embora com 50% de aprovação, teve 37% dos participantes considerando-o difícil, o que pode indicar uma percepção de tédio ou falta de desafio.

Tabela 5.15: Percepção sobre o Balanceamento.

Percepção sobre o Balanceamento			
	PID	**Linear**	**Fácil**
Apropriado	68%	64%	50%
Difícil	9%	11%	37%
Fácil	23%	25%	13%

Fonte: Do autor.

A Tabela 5.16 detalha a distribuição de respostas positivas, neutras, negativas e ambíguas sobre os modos de autobalanceamento. Nota-se que o modo PID teve a menor porcentagem de respostas negativas (18%), enquanto o modo Linear, apesar de 43% de respostas positivas, registrou 29% de avaliações negativas. O modo Fácil apresentou um equilíbrio entre respostas positivas (27%) e negativas (27%).

Tabela 5.16: Quantificação das Respostas sobre o Autobalanceamento.

Percepção do Autobalanceamento			
	PID	**Linear**	**Fácil**
Sem resposta	27%	11%	19%
Positivas	36%	43%	27%
Neutras	14%	11%	23%
Negativas	18%	29%	27%
Ambíguas	5%	6%	4%

Fonte: Do autor.

5.3 Considerações Finais do Capítulo

Este capítulo apresentou a aplicação de um modelo metodológico estruturado para investigar o autobalanceamento em jogos digitais, com base em controladores PID e *Game Analytics*. Dois estudos complementares foram realizados: o estudo exploratório com *InfinityFire* validou a viabilidade técnica da integração PID-Analytics e forneceu um modelo de coleta de dados em tempo real; o estudo experimental com *SpacePilot* aprofundou a análise, comparando os métodos de balanceamento tradicional e PID em diferentes cenários.

Os resultados indicaram que o método PID proporciona ajustes mais dinâmicos e adaptativos em relação ao método linear com correções pontuais de acordo com o *setpoint* determinado. No entanto, não foram identificadas diferenças estatisticamente significativas entre os dois métodos quanto à proporção de acertos e erros, com o PID resultando em uma avaliação de reação melhor com uma diferença sutil.

6. Considerações Finais

6.1 Visão Geral da Pesquisa

Esta dissertação investigou o uso de controladores Proporcional-Integral-Derivativo (PID) integrados a técnicas de *Game Analytics* para autobalanceamento de jogos digitais. Com foco na experiência e no desempenho dos jogadores, foram conduzidos dois estudos complementares: um exploratório com o jogo *InfinityFire* e um experimental com o jogo *SpacePilot*. Esses estudos abordaram questões de pesquisa voltadas à eficácia do balanceamento dinâmico, analisando quantitativamente e qualitativamente o impacto das metodologias propostas. Fundamentada em abordagens experimentais consolidadas, a pesquisa seguiu as etapas de definição, planejamento, execução, análise e interpretação de dados, com base em um modelo sistemático.

6.2 Contribuições e Impactos

Os resultados desta pesquisa indicaram que tanto o método PID quanto o Linear oferecem abordagens adaptativas semelhantes no que diz respeito ao desempenho dos jogadores. Apesar de não terem sido observadas diferenças estatisticamente significativas na proporção de acertos e erros entre os dois métodos, ambos demonstraram eficácia em ajustar a dificuldade do jogo de forma a promover engajamento e aprendizado. A análise quantitativa revelou que os dois métodos apresentaram correlação positiva entre velocidade e pontuação, com o modo Linear exibindo uma relação mais previsível e o PID mostrando ajustes dinâmicos em resposta às variações de desempenho, principalmente em regiões de pontuação fora do *setpoint*, onde apresentam baixas correlações.

As análises qualitativas trouxeram nuances importantes. Participantes relataram percepções positivas tanto para o PID quanto para o Linear, com 68% e 64% considerando os métodos apropriados, respectivamente. Esses dados sugerem que, mesmo sem diferenças significativas em termos de acurácia, ambos os métodos são capazes de oferecer experiências adaptativas satisfatórias para os jogadores.

O método PID, no entanto, destacou-se levemente em termos de experiência percebida. Seu ajuste dinâmico foi descrito como mais responsivo e adaptado às habilidades do jogador, promovendo uma sensação de desafio personalizado. Por outro lado, o modo Linear sugere uma abordagem previsível, com variação gradual dos desafios.

Esses resultados indicam que o PID é uma alternativa viável para sistemas de balanceamento adaptativo, oferecendo diferentes perspectivas de personalização e desafio. A integração com *Game Analytics* foi fundamental para suportar ambas as

abordagens (PID e Linear), permitindo ajustes em tempo real e fornecendo dados valiosos para análise. Embora semelhantes em muitos aspectos, os métodos apresentam características complementares que podem ser exploradas para atender a diferentes perfis de jogadores e objetivos de jogos.

A implementação do experimento proposto neste estudo contribui significativamente para o campo de Sistemas de Informação ao integrar tecnologias avançadas de *Game Analytics* e controladores PID em contextos de jogos digitais. Essa integração demonstra a aplicabilidade de controle e análise de dados em jogos digitais, oferecendo novas perspectivas sobre a otimização de processos de autobalanceamento.

Finalmente, este estudo avança o desenvolvimento de jogos digitais específicos. Ao incorporar elementos de análise de dados e controle PID diretamente no design do jogo, a pesquisa demonstra como jogos podem ser inteligentemente adaptados , essa abordagem pode ser particularmente valiosa para desenvolvedores de jogos interessados em criar experiências que combinem diversão com engajamento.

6.3 Limitações da Pesquisa

Entre as limitações observadas, destaca-se a simplicidade dos jogos utilizados. Embora jogos simples sejam adequados para testes controlados, eles não refletem completamente a diversidade em jogos digitais. A escala reduzida do experimento, com grupos de participantes limitados, também pode restringir a generalização dos resultados.

6.4 Trabalhos Futuros

Para trabalhos futuros, sugere-se ampliar o escopo da pesquisa com jogos mais complexos, abrangendo cenários que representem desafios mais diversificados em jogos. A inclusão de um número maior de participantes, bem como de perfis diversificados, contribuirá para a validação estatística mais robusta dos resultados. Além disso, a implementação de novos métodos de balanceamento, como aprendizado de máquina integrado a controladores PID, pode trazer ganhos adicionais sobre a eficácia de sistemas adaptativos. Por fim, a exploração de aplicações em áreas além de jogos digitais, como treinamentos industriais ou educativos, amplia o potencial de impacto da abordagem proposta.

6.5 Considerações Finais

Esta dissertação contribui significativamente para a pesquisa em jogos digitais e técnicas de balanceamento dinâmico, demonstrando que controladores PID, aliados a *Game*

Analytics, têm potencial para transformar a maneira como jogos são desenvolvidos e aplicados em diferentes contextos. Apesar das limitações observadas, os achados reforçam a viabilidade do autobalanceamento dinâmico PID como ferramenta adaptativa, capaz de personalizar experiências de jogo e potencializar o aprendizado dos usuários. A pesquisa abre caminho para investigações futuras, consolidando uma base metodológica e experimental sólida para o avanço na área de jogos digitais com sistemas adaptativos.

Referências Bibliográficas

ABOELHASSAN, A.; ABDELGELIEL, M.; ZAKZOUK, E. E.; GALEA, M. Design and implementation of model predictive control based pid controller for industrial applications. **Energies**, MDPI, v. 13, n. 24, p. 6594, 2020.

ALTIMIRA, D. *et al.* Enhancing player engagement through game balancing in digitally augmented physical games. **International Journal of Human-Computer Studies**, Elsevier, v. 103, p. 35–47, 2017.

ANDRADE, G.; RAMALHO, G.; SANTANA, H.; CORRUBLE, V. Extending reinforcement learning to provide dynamic game balancing. **Proceedings of the 2006 International Conference on Intelligent User Interfaces (IUI)**, ACM, New York, NY, USA, p. 35–41, 2006.

ARIYANSYAH, Q.; MA'ARIF, A. Dc motor speed control with proportional integral derivative (pid) control on the prototype of a mini-submarine. **Journal of Fuzzy Systems and Control**, v. 1, n. 1, p. 18–24, 2023.

ARMSTRONG, S. Towards game data science: Bridging the gap between game development and data-driven decision making. **International Journal of Computer Games Technology**, v. 2021, p. 1–12, 2021.

BASILI, V. R. **Software modeling and measurement: the Goal/Question/Metric paradigm**. Maryland, 1992.

BASILI, V. R.; CALDIERA, G.; ROMBACH, H. D. The goal question metric approach. **Encyclopedia of Software Engineering**, John Wiley & Sons, Inc., New York, NY, USA, v. 1, p. 528–532, 1992.

CHEUNG, G.; ZIMMERMAN, G. Beyond score: A novel metric of user engagement in digital games. In: **CHI Conference on Human Factors in Computing Systems**. New York: ACM, 2022. p. 1–14.

COHEN, J. **Statistical Power Analysis for the Behavioral Sciences**. 2. ed. Londom: UK: Routledge, 1988.

CRESWELL, J. W.; CRESWELL, J. D. **Research Design: Qualitative, Quantitative, and Mixed Methods Approaches**. 5th. ed. Thousand Oaks, CA: SAGE Publications, 2017. ISBN 9781506386706.

EL-NASR, M. S.; DRACHEN, A.; CANOSSA, A. **Game analytics**. Springer, 2016. Disponível em: <https://link.springer.com/content/pdf/10.1007/978-1-4471-4769-5.pdf>.

FENG, T.; BURNS, E. Game analytics for player progress visualization in platform games. **Entertainment Computing**, v. 33, p. 100340, 2020.

FRUTOS-PASCUAL, M.; ZAPIRAIN, B. G. Review of the use of ai techniques in serious games: Decision making and machine learning. **IEEE Transactions on Computational Intelligence and AI in Games**, IEEE, v. 9, n. 2, p. 133–152, 2015.

GARCIA, A.; PEREIRA, C. Coleta e análise de dados comportamentais em jogos digitais. **Computers in Human Behavior**, Elsevier, v. 112, p. 106438, 2021.

HUNICKE, R.; LEBLANC, M.; ZUBEK, R. *et al.* Mda: A formal approach to game design and game research. In: SAN JOSE. **Proceedings of the AAAI Workshop on Challenges in Game AI**. California, 2004. v. 4, n. 1, p. 1722.

JOHNSON, M.; LEE, A. Cultural significance and economic growth: A study of the gaming industry's rise. **International Journal of Cultural Studies**, Cultural Studies Press, v. 15, n. 2, p. 150–170, 2022.

JOHNSON, M.; WANG, L. Game analytics: Enhancing player engagement through data-driven insights. In: GAME DESIGN SOCIETY. **Proceedings of the International Conference on Game Design**. [S.l.], 2021. p. 150–160.

KEELE, S. *et al.* **Guidelines for performing systematic literature reviews in software engineering**. Keele: UK: Technical report, ver. 2.3 ebse technical report. ebse, 2007.

KENDALL, M. G. A new measure of rank correlation. **Biometrika**, Oxford University Press, v. 30, n. 1-2, p. 81–93, 1938.

KIM, S.-h.; LEE, M.-J. Desafios e avanços no autobalanceamento de jogos eletrônicos. **Journal of Game Development**, Game Dev Publishing, v. 19, n. 3, p. 220–235, 2022.

KITCHENHAM, B.; CHARTERS, S. **Guidelines for performing systematic literature reviews in software engineering**. Keele, 2007.

LEE, J.; JIN, K. Designing a comprehensive game analytics pipeline: From data collection to visualization. **IEEE Transactions on Games**, v. 13, n. 2, p. 205–216, 2021.

LEHMANN, T.; MIRZA-BABAEI, P.; NIESENHAUS, J. Large-scale telemetry data mining in aaa games: Challenges and opportunities. **Entertainment Computing**, v. 44, p. 100511, 2023.

MANN, H. B.; WHITNEY, D. R. On a test of whether one of two random variables is stochastically larger than the other. **Annals of Mathematical Statistics**, v. 18, n. 1, p. 50–60, 1947.

MIDWAY, S. R. Principles of effective data visualization. **Patterns**, Elsevier, v. 1, n. 9, p. 100141, 2020.

MILLER, E.; THOMPSON, R. Engajamento de usuários em jogos sérios: Uma revisão sistemática. **International Journal of Serious Games**, Academic Press, v. 7, n. 1, p. 89–105, 2020.

PEARSON, K. Notes on regression and inheritance in the case of two parents. **Proceedings of the Royal Society of London**, The Royal Society, v. 58, p. 240–242, 1895.

PFAU, J.; LIAPIS, A.; VOLKMAR, G.; YANNAKAKIS, G. N.; MALAKA, R. Dungeons & replicants: automated game balancing via deep player behavior modeling. p. 431–438, 2020.

RECKER, J. **Scientific Research in Information Systems: A Beginner's Guide**. Berlin, Heidelberg: Springer, 2013. ISBN 978-3-642-30047-6.

REIS, A. V. dos. **Artigo: Então você precisa balancear um jogo?** 2016. <https://www.fabricadejogos.net/posts/entao-voce-precisa-balancear-um-jogo/>. [Accessed 23-07-2024].

ROY, D.; SRIVASTAVA, R.; JAT, M.; KARACA, M. S. A complete overview of analytics techniques: descriptive, predictive, and prescriptive. **Decision intelligence analytics and the implementation of strategic business management**, Springer, p. 15–30, 2022.

SANTOS, C. M. d. C.; PIMENTA, C. A. d. M.; NOBRE, M. R. C. A estratégia pico para a construção da pergunta de pesquisa e busca de evidências. **Revista latino-americana de enfermagem**, SciELO Brasil, v. 15, p. 508–511, 2007.

SCHREIBER, I. **Game Balance**. Boston, MA: Course Technology PTR, 2011. ISBN 978-1-58450-711-2.

SERRA, C. B.; CLASSE, T. M. de. Análise e visualização de dados em jogos de treinamento de situações de risco na indústria-um estudo em mapeamento sistemático. 2023.

SHAPIRO, S. S.; WILK, M. B. An analysis of variance test for normality (complete samples). **Biometrika**, Oxford University Press, v. 52, n. 3-4, p. 591–611, 1965.

ŠLOSAR, L. *et al.* Combining physical and virtual worlds for motor-cognitive training interventions: Position paper with guidelines on technology classification in movement-related research. **Frontiers in psychology**, Frontiers Media SA, v. 13, p. 1009052, 2022.

SMITH, J.; DOE, J. Comparative analysis of the economic impact of the gaming, cinematic, and music industries. **Journal of Digital Economics**, Digital Economics Publishing, v. 20, n. 3, p. 200–215, 2023.

SMITH, J.; DOE, J. Growth trends in mobile gaming: An analysis of market dynamics. **Journal of Digital Entertainment**, Digital Publishing, v. 15, n. 2, p. 123–135, 2023.

SPEARMAN, C. The proof and measurement of association between two things. **The American Journal of Psychology**, University of Illinois Press, v. 15, n. 1, p. 72–101, 1904.

SU, Y.; BACKLUND, P.; ENGSTRÖM, H. Comprehensive review and classification of game analytics. **Service Oriented Computing and Applications**, Springer, v. 15, p. 141–156, 2021.

THOMAS, R.; MüLLER, H. Privacidade e autonomia dos jogadores: Implicações Éticas no uso de game analytics. **Ethics in Digital Gaming**, Ethics Publishing, v. 10, n. 4, p. 300–315, 2021.

VILANOVA, R.; VISIOLI, A. The proportionalintegralderivative (pid) controller. In: . Wiley, 2017. Disponível em: <https://api.semanticscholar.org/CorpusID:229303187>.

VILAR, P.; SOUZA, F. Aplicação de controladores pid em sistemas de autobalanceamento para jogos digitais. **Revista de Engenharia de Controle**, Instituto de Engenharia, v. 34, n. 2, p. 98–112, 2019.

VOLKMAR, L.; DAHLSKOG, S. A systematic literature review of player behavior analysis in online games. In: IEEE. **2020 IEEE Conference on Games (CoG)**. New York, 2020. p. 1–8.

WANG, Q.; LI, Y.; CHEN, C.; SONG, X. Predictive modeling for player retention using neural networks and behavioral features. **Expert Systems with Applications**, v. 188, p. 116009, 2022.

WOHLIN, C. *et al.* **Experimentation in Software Engineering**. Pennsylvania: Springer, 2012.

XIA, W.; ANAND, B. Game balancing with ecosystem mechanism. In: IEEE. **2016 international conference on data mining and advanced computing (SAPIENCE)**. New York, 2016. p. 317–324.

YANG, Q. *et al.* Adaptive difficulty adjustment using players performance and preferences. **International Journal of Human-Computer Studies**, Elsevier, v. 105, p. 61–70, 2017.

APÊNDICE A – Título

.1 Mapeamento Sistemático da Literatura

.1.1 Planejamento

O planejamento em um MSL é determinado a partir da confecção de um protocolo a ser seguido para a condução do estudo. Assim, o protocolo deste MSL envolveu as etapas: i) definição do objetivo, ii) elaboração das questões de pesquisa, iii) seleção das fontes de dados, iv) composição da *string* de busca, v) definição de critérios de inclusão, exclusão e avaliação da qualidade dos trabalhos, vi) Aplicação da técnica de snowballing (backward e forward), a fim de ampliar a seleção de estudos a partir das referências e citações dos artigos identificados.

O objetivo foi definido com base no problema de pesquisa **Como jogos digitais utilizam técnicas de autobalanceamento?** O protocolo foi estruturado de acordo com a abordagem GQM (*Goal Question Metrics*) (Basili, 1992), sendo definido como: **analisar** a existência de estudos primários; **com o propósito de** identificar técnicas e conceitos de autobalanceamento; **relacionados a** como são usados, aplicados e implantados; **do ponto de vista de** pesquisadores; **no contexto de** jogos digitais.

Com base neste objetivo, foram elaboradas **5 questões secundárias de pesquisa**:

Q1: Que técnicas de autobalanceamento são utilizadas em jogos digitais?

Q2: Que tipo de dados são coletados para o autobalanceamento do jogo criado?

Q3: Quais ferramentas foram utilizadas para validar a eficácia da utilização do autobalanceamento?

Q4: Quais as mecânicas do jogo criado que são influenciadas pelo balanceamento?

Q5: Qual tipo de jogo foi criado?

Para a **seleção das fontes de dados**, foram utilizadas as principais fontes de dados onde estudos de computação e trabalhos de jogos são indexados (Keele *et al.*, 2007): *ACM*

Digital Library[1], *EI Compendex*[2], *IEEE Digital Library*[3], *Scopus*[4], *ISI Web of Science*[5] e *Springer*[6].

A partir disso, a **estratégia de busca** foi definida a partir do uso do PICOC [7] (População, Intervenção, Comparação, Saída e Contexto) (Santos *et al.*, 2007), o qual foi utilizado para organização e composição da *string* de busca nas bases (Tabela 1).

Tabela 1: Termos e sinônimos usados no PICOC

Dimensão	Termo em Português	Termo em inglês
População	Jogos Digitais	*Digital Game*
Intervenção	Autobalanceamento	*"balance"OR "balancing"OR "Dynamic Diffucult Adjustment"OR "Dynamic Adjustment"*

Ao criar a *string* de busca, foram incluídos sinônimos das palavras-chave em inglês. Dessa maneira, a *string* de busca foi formulada como:

("Digital game")
AND
("DDA" OR "Dynamic Difficulty Adjustment")

Devido às fontes e resultados variados, foram **definidos critérios de inclusão e exclusão** (Tabela 2). Desta maneira, estudos que apresentassem CI deveriam ser selecionados e analisados nas etapas de seleção e leitura completa (aceitação). Os estudos que forem classificados em ao menos um CE, deveriam ser excluídos das análises.

Tabela 2: Critérios de Inclusão e Exclusão

Código	Descrição.
CI-1	Estudo que aborde o uso de autobalanceamento em jogos digitais.
CE-1	Estudo com acesso indisponível para visualização completa nas bases de dados científicas.
CE-2	Estudo com menos de 4 páginas.
CE-3	Estudo duplicado.
CE-4	Estudo que não aborde o uso de autobalanceamento em jogos digitais.
CE-5	Estudo que não seja primário.
CE-6	Estudo que não esteja escrito nos idiomas português ou inglês.
CE-7	Estudo que seja prefácio, livro, editorial, resumo, pôster, painel, palestra, mesa redonda, oficina, keynotes, tutoriais ou demonstração.

Para a etapa de aceitação foram estabelecidos 6 critérios de qualidade que deveriam ser observados na extração dos dados (Tabela 3). A estes critérios foram atribuídas notas para obter uma pontuação final. Neste trabalho, as notas variavam de 0 a 1, indicando se o estudo atendia ou não a cada critério específico.

Durante a condução do MSL, o *Parsif.al*[8] foi utilizado como uma ferramenta de apoio para o planejamento e organização das referências encontradas na etapa de busca. A

[1] <https://dl.acm.org/>
[2] <http://www.engineeringvillage.com>
[3] <http://ieeexplore.ieee.org>
[4] <http://www.scopus.com>
[5] <http://www.isiknowledge.com>
[6] <http://link.springer.com>
[7] Apenas utilizado População e Intervenção, pois com uma maior delimitação da estratégia de busca, não houve resultado nas buscas
[8] <https://parsif.al/>

partir disso, as análises foram organizadas em 3 etapas: remoção de duplicatas (etapa 1), seleção dos estudos (etapa 2) e aceitação dos trabalhos (etapa 3) que atendiam aos critérios de inclusão. O sistema *Parsif.al* auxiliou, principalmente, na etapa inicial de planejamento e organização das referências porém, o *Microsoft Excel* foi utilizado para a organização e análise dos estudos durante as etapas de seleção e aceitação.

Avaliação Pesquisa Mestrado - Cristiano Serra

Termo de Consentimento Livre e Esclarecido

Esta avaliação faz parte do trabalho para Dissertação de Mestrado do aluno **Cristiano Barroso Serra,** aluno do curso de Informática - Mestrado Acadêmico da Universidade Federal do Estado do Rio de Janeiro (UNIRIO), orientado pelo professor Dr. **Tadeu Moreira de Classe**.

Esta avaliação tem como base a Resolução 510/16, que versa sobre a possibilidade de execução de pesquisas de opinião, é apresentando o projeto de uma proposta para **balanceamento de jogos digital usando controlador PID (Proporcional-Integrativo-Derivativo)**.

O objetivo do estudo consiste em avaliar a dificuldade percebida pelo jogador durante o gameplay. Isto é, a percepção se "como" ou "se" o jogo se adapta as habilidades do jogador durante o jogo. Salientamos que o jogo irá capturar informações de velocidade e pontuação durante o gameplay. Nenhuma informação sensível (nome, etc.) do jogador é coletada.
O estudo é composto de 3 (três etapas):

1. A primeira etapa, iremos identificar o perfil do nosso participantes. Assim deverá ser respondido uma parte inicial do questionário para tal.
2. Na segunda etapa o participante executará executar o jogo em questão durante todo o tempo do jogo. **É necessário que jogue até o final, independente se está muito fácil o difícil**.
3. Por último, o jogador deverá responder um questionário, com poucas questões sobre a percepção de dificuldade do jogo.

O participante, a qualquer momento, tem o direito de se recusar a participar desse estudo e/ou retirar o consentimento de participação caso deseje não mais participar da pesquisa. Neste caso, o participante não sofrerá qualquer prejuízo quanto à sua relação com os pesquisadores.

A participação é facultativa, privada e sigilosa, ou seja, nenhuma informação quanto ao nome, e-mail ou qualquer informação que possa revelar a identidade do participante será coletada, assim com versa a Lei Geral de Proteção de Dados (LGPD).

Pesquisadores envolvidos nesse projeto: Cristiano Barroso Serra (BSI/UNIRIO) e professor Tadeu Classe (UNIRIO). Será possível manter contato do e-mail: cristianoserra@edu.unirio.br.

É assegurada a assistência durante toda pesquisa, bem como é garantido o livre acesso a todas as informações e esclarecimentos adicionais sobre o estudo e suas consequências, enfim, tudo o que queira saber antes, durante e depois da participação.

Enfim, tendo sido orientado quanto ao teor de tudo aqui mencionado e compreendido a natureza e o objetivo do já referido estudo, manifesto meu livre consentimento em

participar, estando totalmente ciente de que não há nenhum valor econômico, a receber ou a pagar, por minha participação.

Em caso de reclamação ou qualquer tipo de dúvida sobre este estudo devo escrever para: cristianoserra@edu.unirio.br.

Rio de Janeiro, 03 de dezembro de 2024.

Aluno, Cristiano Barroso Serra (UNIRIO)

Prof. Dr. Tadeu Moreira de Classe

* Indica uma pergunta obrigatória

1. Declaro que li, entendi e aceito participar deste estudo. *

   Marcar apenas uma oval.

   - ◯ Sim *Pular para a pergunta 2*
   - ◯ Não

Perfil do Participante

Queremos conhecer melhor o perfil do participante do estudo.

2. Idade *

_____

3. Grau de formação *

Marcar apenas uma oval.

- ◯ Fundamental Incompleto
- ◯ Fundamental Completo
- ◯ Médio Incompleto
- ◯ Médio Completo
- ◯ Superior Incompleto
- ◯ Superior Completo
- ◯ Mestrado
- ◯ Doutorado

4. Estado de Residência *

Marcar apenas uma oval.

- AC
- AL
- AM
- AP
- BA
- CE
- DF
- ES
- GO
- MA
- MG
- MS
- MT
- PA
- PB
- PE
- PI
- PR
- RJ
- RN
- RO
- RR
- RS
- SC
- SE
- SP
- TO

5. Gênero *

Marcar apenas uma oval.

- ◯ Masculino
- ◯ Feminino
- ◯ Outro: _____

6. Com que frequência costuma jogar jogos digitais? *

Marcar apenas uma oval.

- ◯ Nunca: nunca jogo.
- ◯ Raramente: jogo de tempos em tempos.
- ◯ Mensalmente: jogo pelo menos uma vez por mês.
- ◯ Semanalmente: jogo pelo menos uma vez por semana.
- ◯ Diariamente: jogo todos os dias.

7. Com base em seu perfil de jogador, você prefere jogos: *

Marcar apenas uma oval.

- ◯ Fáceis
- ◯ Difíceis
- ◯ Que se adaptam as minhas habilidades

VAMOS JOGAR!

Atenção:

1. O jogo pedirá um nome de usuário (apelido, nick, alias) assim que você começar a jogar. Por favor, para lhe manter como um participante anônimo, pedimos que **não coloque o seu nome.** Você pode entrar com qualquer apelido que desejar.
2. Caso você jogue no computador, você guiará a nave com o mouse, segurando o botão esquerdo. Caso jogue no celular ou mobile, a nave será guiada arrastando seu dedo na tela.
3. Para podermos relacionar os dados do gameplay às respostas deste questionário, informe abaixo um apelido. **O MESMO DEVERÁ SER USADO POR VOCÊ NO JOGO**.
4. Bom jogo! Divirta-se.

8. Qual o apelido você vai usar no jogo? *

_____

Acessar o Jogo
OBS: na versão **MOBILE** (jogada no celular), acontece um erro de *JavaScript*, mas é só apertar "OK" que o jogo funciona normalmente. O mesmo não acontece na versão do navegador no computador.

ACESSE: [ACESSE O JOGO CLICANDO AQUI]
Caso ocorra mensagens de erro, clique em ok - isso não impactará no jogo.

9. **Terminou de jogar?*** *

É muito importante que você jogue o jogo até o final antes de ir para a próxima parte do estudo. Do contrário, o estudo será prejudicado. Contamos com a a sua colaboração.

Marcar apenas uma oval.

- ◯ Sim *Pular para a pergunta 10*
- ◯ Não *Pular para a pergunta 8*

Percepção de Dificuldades

10. Com relação a dificuldade geral do jogo, eu considero que este jogo foi: *

Marcar apenas uma oval.

- ◯ Fácil
- ◯ Apropriada
- ◯ Difícil

11. Sobre o balanceamento do jogo, foi possível perceber, que a velocidade... *

Marcar apenas uma oval.

- ◯ ... não percebi alteração na velocidade.
- ◯ ... percebi uma alteração linear (a velocidade alterada na mesma proporção de acertos e erros)
- ◯ ... percebi uma alteração dinâmica (a velocidade alterava de acordo com meus acertos e erros, porém, não em uma taxa linear)

12. Descreva sua percepção em relação ao balanceamento de dificuldade do jogo.

Tabela 3: Critérios de Qualidade

Código	Descrição
CQ1	O estudo responde alguma das questões de pesquisa?
CQ2	O objetivo do estudo está definido de forma clara?
CQ3	Existe secção de trabalhos relacionados, no qual diferencia a proposta do estudo de outros trabalhos?
CQ4	A metodologia da pesquisa é claramente definida e apresentada?
CQ5	Os resultados estão claramente apresentados?
CQ6	O estudo apresenta claramente o contexto de treinamento para qual o jogo foi aplicado?

APÊNDICE B – Exemplo

Código Python utilizado para analisar correlações e distribuições entre duas variáveis (X e Y). O código verifica a normalidade das variáveis e sugere o método de correlação mais adequado com base nos resultados.

```
# Dados
XData = (20.0,20.0,22.0,29.0)
YData = (1,1,1,1)

# Scatter plot para visualizar a relação
plt.scatter(XData, YData)
# plt.title("Scatter Plot")
plt.xlabel("X Speed")
plt.ylabel("Y Score")
plt.show()

# Analisando e recomendando o método de correlação
def analyze_correlation_and_distribution(x, y):
    # Teste de normalidade
    shapiro_x = shapiro(x)
    shapiro_y = shapiro(y)
    is_x_normal = shapiro_x.pvalue > 0.05
    is_y_normal = shapiro_y.pvalue > 0.05

    # Correlações
    pearson_corr, _ = pearsonr(x, y)
    spearman_corr, _ = spearmanr(x, y)
    kendall_corr, _ = kendalltau(x, y)

    # Verificar a normalidade dos dados
    if is_x_normal and is_y_normal:
        distribution_type = "ambas as variáveis seguem uma
            distribuição normal"
        best_correlation = f"Correlação de Pearson é mais
```

```
            adequada (valor: {pearson_corr:.2f})."
    else:
        distribution_type = "uma ou ambas as variáveis não
            seguem uma distribuição normal"
        # Comparar Spearman e Kendall
        if abs(spearman_corr) > abs(kendall_corr):
            best_correlation = f"Correlação de Spearman é mais
                adequada (valor: {spearman_corr:.2f})."
        else:
            best_correlation = f"Correlação de Kendall Tau é
                mais adequada (valor: {kendall_corr:.2f})."

    # Resumo da análise
    result = (
        f"Com base nos testes de normalidade:\n"
        f"- X {'é normal' if is_x_normal else 'não é normal'} (p
            -valor: {shapiro_x.pvalue:.4f}).\n"
        f"- Y {'é normal' if is_y_normal else 'não é normal'} (p
            -valor: {shapiro_y.pvalue:.4f}).\n\n"
        f"Tipo de distribuição: {distribution_type}.\n"
        f"{best_correlation}\n"
    )
    return result

# Chamar a função e exibir os resultados
response = analyze_correlation_and_distribution(XData, YData)
print(response)
```