Desenvolvimento de jogos: Uma Abordagem Prática com Unity e Scrum

Cleydson P. da Silva[1], **Ermeson Andrande**[1], **Julian Araújo**[2], **Danilo Ricardo**[1]

[1]Departamento de Computação
Universidade Federal Rural de Pernambuco (UFRPE) – Recife – PE – Brasil

[2]Centro de Ciências e Tecnologia
Universidade Federal do Cariri (UFCA) – Juazeiro do Norte - CE - Brasil

{cleydson.silva,ermson.andrade, danilo.araujo}@ufrpe.br, carlos.julian@ufca.edu.br

Abstract. *This paper presents an experience report on the development of a survival horror game using the Unity engine, exploring the technical and practical challenges encountered in the process. The project combines theoretical and practical knowledge from Computer Science across various areas, from programming and design to project management. The Scrum methodology was adopted to organize tasks, and tools such as Blender, GIMP, Audacity, and Jira were used in conjunction with Unity. The article discusses the main difficulties faced, the solutions implemented, and the lessons learned throughout the process, offering a valuable reference for students and future game developers.*

Resumo. *Este trabalho apresenta um relato de experiência sobre o desenvolvimento de um jogo de survival horror utilizando a engine Unity, explorando os desafios técnicos e práticos encontrados no processo. O projeto combina conhecimentos teóricos e práticos da Ciência da Computação em diversas áreas, desde programação e design até gerenciamento de projetos. A metodologia Scrum foi adotada para organizar as tarefas, e ferramentas como Blender, GIMP, Audacity e Jira foram utilizadas em conjunto com a Unity. O artigo discute as principais dificuldades enfrentadas, as soluções implementadas e os aprendizados adquiridos, oferecendo uma referência valiosa para estudantes e futuros desenvolvedores de jogos.*

1. Introdução

O Brasil figura como o terceiro maior mercado consumidor de jogos eletrônicos, conforme dados do portal gov.br [Ministério da Ciência, Tecnologia e Inovação 2023]. No entanto, a produção nacional de jogos eletrônicos ainda se encontra em desenvolvimento, com apenas pequenos e médios estúdios alcançando destaque [Associação Brasileira das Desenvolvedoras de Jogos Digitais (Abragames) 2023]. A carência de incentivos para o setor limita o potencial do país em um dos mercados mais promissores globalmente. Desde 2008, a indústria de jogos eletrônicos supera em lucratividade os setores de cinema e música combinados, conforme reportado pela Veja [Veja 2023]. A popularização dos jogos online e mobile expandiu o público consumidor, antes concentrado em consoles e com putadores pessoais, como evidenciado em matéria do portal correio [Portal Correio 2025].

Além do entretenimento, os jogos eletrônicos oferecem aplicações em simuladores para treinamento profissional, conforme informações do Instituto Claro [Instituto Claro 2023]. Adicionalmente, softwares de modelagem 3D beneficiam áreas como arquitetura, medicina e engenharia mecânica. As *gamefication* que adiciona mecânicas de jogo a atividades do mundo real para torná-las mais envolventes, estão cada vez mais ganhando espaço no ambiente escolar, profissional e até na médicina, como aponta o artigo [Maria Carolina Ortiz Whitaker 2021] e [José Hícaro Hellano 2019].

O desenvolvimento de jogos, mesmo em projetos de menor escala, representa um desafio complexo, demandando conhecimentos em diversas áreas, incluindo programação, lógica, linguagens de programação, modelagem 3D, design, arquitetura e física, projetos em equipe exigem gerenciamento eficaz de recursos e pessoal, visando a coordenação do trabalho dentro dos prazos e ferramentas disponíveis. Diante desse contexto, emerge a seguinte questão de pesquisa: ”Quais desafios técnicos e práticos são encontrados no desenvolvimento de um jogo, e como a aplicação dos conhecimentos teóricos e práticos da Ciência da Computação pode contribuir para a solução desses desafios?”. A investigação busca analisar e relatar o processo de desenvolvimento de um jogo como projeto integrador, identificando dificuldades, ferramentas e aprendizados decorrentes da aplicação prática dos conteúdos do curso de Ciência da Computação.

Em face de um cenário pouco estimulado no Brasil e de um processo de criação inerentemente complexo, este trabalho pode servir como referência prática para estudantes interessados em conhecer as etapas de desenvolvimento de jogos, suas dificuldades e soluções para problemas específicos, abordando também aspectos como a seleção e o gerenciamento de ferramentas e equipes. A aplicação de conceitos teóricos em um projeto real proporciona uma visão abrangente das etapas de desenvolvimento, ferramentas e desafios, especialmente para iniciantes, podendo inspirar o desenvolvimento de projetos similares ou o interesse pelo mercado de desenvolvimento de jogos.

Este trabalho possui natureza descritiva, exploratória e reflexiva, adotando uma abordagem qualitativa. Será relatada a experiência no desenvolvimento de um jogo, analisando os processos, desafios técnicos, seleção e uso de ferramentas, soluções, gerenciamento de equipe e escopo, e aprendizados. A abordagem pessoal permitiu uma análise detalhada de aspectos subjetivos do processo, avaliando o processo de análise e tomada de decisões à luz dos conhecimentos adquiridos na formação.

A estrutura deste trabalho é a seguinte: a Seção 2 apresenta a fundamentação teórica, abordando os conceitos sobre as ferramentas e mecânicas utilizadas na produção do jogo. A Seção 3 descreve os métodos de gerênciamento de tarefas, a Seção 4 descreve a implementação de várias recursos presente no jogo. Por fim, a Seção 5 apresenta as experiências, discute as conclusões e propõe trabalhos futuros.

2. Fundamentação Teórica

2.1. Importância da indústria de jogos

A indústria de jogos eletrônicos se destaca como um dos setores mais lucrativos do entretenimento global. O crescimento exponencial do consumo de jogos mobile na última década solidifica sua relevância, especialmente em mercados como os Estados Unidos e a Europa, onde gera milhares de empregos, conforme apontado nos estudos de Emanoel

Querette [do artigo 2013] e Alexandre Cesar [Ribeiro 2015] sobre o potencial econômico da indústria de jogos.

No Brasil, apesar do crescimento promissor, o setor ainda enfrenta desafios para alcançar seu pleno potencial, como infraestrutura limitada, incentivos governamentais reduzidos e dificuldades de captação de investimentos.

Além do entretenimento, a indústria de jogos eletrônicos também impulsiona avanços tecnológicos em áreas como inteligência artificial, ambientes virtuais realistas e simuladores para treinamento em diversos setores, conforme abordado em reportagem do blog Atlântico [Atlântico 2023].

2.2. Jogos AAA e indies

Jogos AAA são grande produções, desenvolvidos por grandes estúdios tendo a disposição elevado orçamento e equipes de desenvolvimento compostas por centenas de pessoas. Esses jogos geralmente possuem gráficos e mecânicas realistas e complexos. Exemplos de jogos AAA incluem títulos como The Witcher 3, Red Dead Redemption 2 e Call of Duty. Por outro lado, jogos indie (independentes) são desenvolvidos por equipes pequenas ou até por uma única pessoa, com orçamentos mais modestos ou até inexistente, porém com grande liberdade criativa. Esses jogos costumam explorar mecânicas inovadoras e abordagens únicas de design e gráficos. Exemplos notáveis incluem Hollow Knight, Undertale e Stardew Valley. A principal diferença entre eles é que, enquanto os jogos AAA visam um grande público com produções de alto custo, os jogos indie se destacam pela originalidade, pelo custo mais acessível e pela jogabilidade mais descompromissada, embora também existam títulos indie com uma complexidade impressionante.

2.3. Jogos 2Ds e 3Ds

Jogos 2D têm sua perspectiva definida apenas em dois eixos: X e Y. Todos os objetos, personagens, cenários e interfaces são representados em duas dimensões. As movimentações possíveis são limitadas para direita, esquerda, cima e baixo. Jogos 2D são amplamente reconhecidos pelos gêneros de plataforma, RPGs e jogos de escolhas. Jogos 3D adicionam o eixo da profundidade (Z), criando cenários mais próximos ao mundo real. Os objetos e personagens ganham profundidade e podem ser vistos de vários ângulos. A movimentação é possível em todas as direções. Jogos como FPS (*first-person shooters*) e TPS (*third-person shooters*) são exemplos de gêneros presentes nos jogos 3D. Inicialmente, os jogos eram essencialmente 2D devido às limitações de hardware.

Na década de 90, a revolução dos jogos 3D teve início com ferramentas e consoles mais potentes, que possibilitaram o desenvolvimento de jogos mais complexos, com o uso da perspectiva de profundidade. Com o tempo, jogos 2D passaram a ser mais associados a um estilo artístico, ao invés de uma limitação técnica. Alguns jogos 2D tentaram simular o 3D, como Doom (1993), que, embora utilizasse gráficos 2D, oferecia uma sensação de profundidade com seus ambientes ”falsos 3D”, e Top Gear, um jogo de corrida que simulava uma pista em 3D. Jogos 2,5D são aqueles que limitam a movimentação do jogador aos eixos 2D, mas que possuem ambientes ou elementos construídos em 3D, criando uma ilusão de profundidade. Exemplos incluem Street Fighter IV e New Super Mario Bros. Hoje, jogos 2D e 3D são expressões artísticas, e muitos títulos buscam brincar com a fusão desses estilos. O projeto deste artigo elabora cenários que buscam integrar elementos 2D e 3D, criando uma experiência visual híbrida.

2.4. FPS (*first-person shooters*)

O projeto deste artigo se insere na categoria de FPS (*First-Person Shooter*). Este gênero é caracterizado pela perspectiva em primeira pessoa e pelo combate, geralmente com armas de fogo, embora também possa incluir armas brancas em alguns jogos. Exemplos desse gênero incluem *Doom, Blood, Call of Duty* e *Battlefield.*

Existem outros gêneros que compartilham semelhanças com o FPS, como o walking simulator, que também utiliza a perspectiva em primeira pessoa, mas se diferencia por focar na exploração e narrativa, sem combate. O objetivo é proporcionar uma experiência imersiva no ambiente e na história, frequentemente sem ações violentas.

Outro gênero frequentemente confundido com o FPS é o escape game, que também adota a visão em primeira pessoa. Porém, ao contrário do FPS, os escape games não envolvem combate, mas se concentram em resolver quebra-cabeças ou escapar de situações perigosas, com o jogador utilizando o cenário a seu favor.

2.5. Survival Horror

Focado em criar uma experiência de medo, tensão e suspense, o survival horror geralmente se passa em ambientes claustrofóbicos e amedrontadores. Os inimigos normalmente são monstros, e o jogador possui poucos recursos, como munição e itens de cura. A jogabilidade se divide entre explorar cenários para coletar recursos, combater inimigos, fugir e resolver puzzles. A história, geralmente, foca na resolução de mistérios. Esses aspectos são analisados de forma mais detalhada no artigo de Bruno Felippe [Barbosa 2015].

Além de ser um FPS, o projeto Sobreviva também se enquadra na categoria de survival horror. Jogos como Resident Evil e Silent Hill foram referências importantes, especialmente Resident Evil 7, lançado em 2016 pela Capcom, que serviu de base para a construção do sistema de combate, inventário e ambientação deste projeto.

2.6. Combinações de gêneros

Existem diversos gêneros no mundo dos jogos eletrônicos, como jogos de esportes, jogos focados em escolhas e narrativas, jogos de terceira pessoa, entre outros. Além dos gêneros já mencionados, também existem jogos educativos, que utilizam elementos de gamificação.

Jogos eletrônicos podem ter mais de um gênero, com um deles sendo dominante. Resident Evil 7, por exemplo, é essencialmente um survival horror, mas, por apresentar combate em primeira pessoa, também é considerado um FPS. Além disso, há jogos que permitem ao jogador alternar entre diferentes perspectivas durante o jogo, como PUBG, que oferece as opções de primeira e terceira pessoa

2.7. Eficácia dos survival horror

Os jogos eletrônicos devem incorporar elementos que mantenham o jogador envolvido e se tornem divertidos, já que seu principal objetivo é entreter. No entanto, a definição de um jogo bom depende muito do gênero, pois cada tipo de jogo é direcionado a um público específico. Por exemplo, um jogo que se propõe a oferecer uma experiência narrativa deve ser capaz de criar uma história com personagens interessantes para o público-alvo.

Já jogos online devem focar em uma jogabilidade fluida e em mapas eficazes para tornar as partidas dinâmicas e interessantes.

Cada jogo tem sua proposta, e no gênero de survival horror, os elementos mais marcantes estão no level design. Os mapas e a composição dos cenários são, por si mesmos, um personagem do jogo. Elementos de luz e sombra, conexões entre os mapas, progressão e exploração são chave para um bom jogo desse gênero. Exemplos de jogos aclamados por explorarem um bom level design são o Resident Evil 2 Remake e Silent Hill 2 Remake. Por outro lado, jogos que, apesar de apresentarem qualidades gráficas e boa jogabilidade, não foram bem recebidos justamente pela exploração dos mapas incluem The Callisto Protocol (lançado em 2022) e Resident Evil 3 Remake. Seus mapas eram simples e monótonos, o jogador seguia em linha reta, sem a necessidade de b*acktracking* (voltar aos mesmos trechos do mapa para liberar novas áreas e itens).

Além de uma boa exploração, o combate deve ser cadenciado. O jogador não pode se sentir nem muito forte nem muito fraco, para que a sensação de tensão esteja sempre presente, mas sem se tornar frustrante. Uma boa jogabilidade é fundamental para que executar as ações no jogo seja prazeroso e o jogador não se sinta injustiçado caso falhe devido a um defeito de jogabilidade.

Por fim, mas não menos importante, a história deve cativar o sentimento de descoberta. Normalmente, no gênero survival horror, o mundo está em caos ou é estranho, e o jogador deve descobrir o motivo de estar ali e o que está acontecendo.

2.8. Etapas de desenvolvimento de um jogo

O desenvolvimento de um jogo passa por diversas etapas e depende muito do seu público-alvo, estilo gráfico e gênero. No entanto, algumas fases do processo são comuns a todos os tipos de jogos, independentemente da complexidade ou da plataforma.

A primeira etapa é a idealização, onde se define o conceito central do projeto: do que se trata o jogo, para quem ele é destinado, quais serão os elementos mais chamativos e como o jogador será imerso na experiência. Essa fase pode surgir de uma demanda de um estúdio para atrair um público específico ou de uma ideia central que, com o tempo, se desenvolve para abranger todo o universo do jogo.

A segunda etapa é a concepção, onde a história e as ideias iniciais do jogo são mais detalhadas. Durante essa fase, são elaborados materiais gráficos como esboços de personagens, conceitos de cenários, rascunhos de mapas e uma definição inicial das mecânicas do jogo. Essa etapa é crucial para dar uma direção clara ao projeto, garantindo que todos os elementos e funcionalidades estejam alinhados com a proposta do jogo. As mecânicas também começam a ser esboçadas aqui, com uma visão inicial do comportamento do jogo.

Após a elaboração das ideias e rascunhos, os desenvolvedores avançam para a construção dos assets. Os assets são todos os elementos visuais, sonoros e funcionais do jogo. Cada asset é projetado separadamente, utilizando ferramentas especializadas, mas todos precisam ser integrados de forma coerente para que o jogo tenha uma aparência e funcionamento consistentes.

A etapa seguinte envolve a integração desses assets em um motor gráfico, como discutido na seção *Motores de Jogo: A Base do Desenvolvimento*. O motor gráfico é a

ferramenta que unifica todos os componentes do jogo e permite que os desenvolvedores testem e ajustem a jogabilidade, o design visual e a física do jogo. Durante essa fase, os desenvolvedores ajustam os assets, testam as mecânicas e refinam o jogo para garantir que todos os elementos funcionem de forma harmônica e fluida.

2.9. Motores de Jogo: A Base do Desenvolvimento

Os motores de jogo são softwares que fornecem um conjunto de ferramentas para auxiliar no desenvolvimento de jogos eletrônicos e simuladores virtuais [Salutes 2023]. Eles integram funcionalidades para o gerenciamento de gráficos, física, som e lógica do jogo, permitindo a criação com alta produtividade de cenários detalhados, animações realistas e comportamentos interativos complexos.

Atualmente, há uma ampla variedade de motores de jogo disponíveis, desde aqueles desenvolvidos por grandes empresas até alternativas de código aberto, como Unreal Engine e Unity [NetCoders 2023]. Para este projeto, optou-se pelo Unity devido ao conhecimento prévio da equipe, à vasta quantidade de tutoriais disponíveis e à compatibilidade com os recursos técnicos e financeiros disponíveis.

3. Metodologia de Desenvolvimento

3.1. Abordagem Metodológica: Scrum

O projeto foi desenvolvido em equipe, com cada membro focado em uma parte específica do desenvolvimento. As funções foram divididas entre a criação de mecânicas e scripts, e a elaboração de todos os aspectos visuais, como modelagem 3D, animação 2D e composição de cenários. Toda a equipe discutia a história, as descrições das mecânicas e a implementação de novos recursos.

Para o gerenciamento eficaz do projeto, a equipe adotou a metodologia Scrum, um framework ágil, iterativo e incremental. A escolha do Scrum foi motivada por sua flexibilidade e capacidade de adaptação a projetos complexos, permitindo entregas incrementais e melhorias contínuas ao longo do desenvolvimento.

3.2. Ferramentas de Gerenciamento: Jira e *Backlog*

A ferramenta Jira foi utilizada para a criação e gestão do backlog do projeto. O *backlog* foi estruturado em cards, cada um representando uma tarefa específica, categorizadas em:

- **Aspectos Visuais**: Modelagem 3D, texturização, animação e design de interface.
- **Mecânicas e Programação**: Desenvolvimento de scripts, implementação de funcionalidades e lógica do jogo.
- **Documentação**: Criação de documentação técnica e manual do usuário.
- **Planejamento**: Definição de metas, cronograma e alocação de recursos.

3.3. Ciclos de Desenvolvimento: Sprints e Retrospectivas

O desenvolvimento foi organizado em sprints de duas semanas, com reuniões de retrospectiva ao final de cada ciclo. As retrospectivas, realizadas online, tinham como objetivos:

- Avaliar o progresso da sprint atual.
- Identificar pontos positivos e negativos do processo.

- Discutir impedimentos e soluções.
- Revisar e validar os cards concluídos.
- Planejar a próxima sprint, definindo novas tarefas e prioridades.

O fluxo de sprint é resumido na Figura 1. Ciclos de duas semanas entre o início e término da sprint, com a retrospectiva para iniciar uma nova sprint.

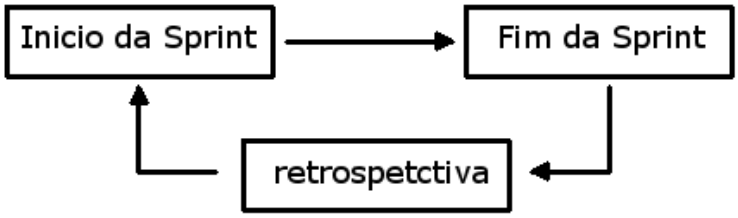

Figura 1. Ciclos de sprints.

Durante os sprints, a equipe mantém comunicação constante para alinhamento, resolução de dúvidas e acompanhamento das atividades. Eram atribuídas responsabilidades específicas a cada integrante. Usando a ferramenta *Git*, que será explicada na próxima seção (3.4), foram criadas duas branches: uma para aspectos gráficos e outra para a criação de scripts e mecânicas. No fim da sprint, era feita uma revisão das atividades realizadas; aquelas que estavam estáveis eram incorporadas à branch *main*.

- Um desenvolvedor focou na criação e implementação de scripts e funcionalidades do jogo.
- O outro desenvolvedor dedicou-se à criação dos aspectos visuais e sonoros.

3.4. Controle de Versão e Colaboração: *Git* e *GitHub*

Para o controle de versão do código-fonte e a colaboração entre os membros da equipe, foram utilizadas as ferramentas *Git* e *GitHub*. O projeto foi hospedado no GitHub, permitindo o gerenciamento de diferentes versões do código e a criação de branches para o desenvolvimento paralelo de funcionalidades.

3.5. Ferramentas Utilizadas no Desenvolvimento

- **Unity:** Motor de jogo escolhido para o desenvolvimento. A escolha se baseou no conhecimento prévio adquirido no componente curricular motores gráficos e na grande quantidade de material e tutoriais disponíveis online. Além disso, seu site oficial oferece documentação extensa sobre a ferramenta [Unity Technologies 2023]. O Unity também foi escolhido por ter um modelo de licenciamento mais acessível para a equipe.
- **Blender:**
  Software de modelagem e animação 3D utilizado para criar modelos, mapeamento UV e animações do jogo [Blender Foundation 2023]. Foi escolhido devido à sua versatilidade, recursos avançados e familiaridade da equipe com a ferramenta.
- **Gimp:**
  Editor de imagens 2D utilizado para a criação de texturas e animações 2D de personagens e elementos do jogo [GIMP Team 2023]. Trata-se de um software

leve e de fácil aprendizado, utilizado na elaboração de texturas e sprites animados para o jogador, inimigos e demais elementos gráficos do jogo.

- **Audacity:** Editor de áudio utilizado para a edição e aprimoramento de sons ambientes, efeitos sonoros e trilha sonora do jogo [Audacity Team 2023]. A ferramenta foi usada para melhorar a qualidade dos áudios implementados no jogo.
- **VSCode:** Editor de código-fonte utilizado para programação em C, a principal linguagem do Unity, com o auxílio de extensões que otimizam o desenvolvimento [Microsoft 2023].
- **Git e GitHub:**
  Sistema de controle de versão utilizado para o gerenciamento do código-fonte, permitindo desenvolvimento simultâneo em diferentes aspectos do jogo [GitHub 2023]. O projeto foi hospedado no GitHub devido à sua facilidade de compartilhamento e controle de versões. Foram criadas duas branches principais: uma para o desenvolvimento visual e outra para as mecânicas do jogo, além de uma branch master para integração periódica.
- **Jira:** Ferramenta de gerenciamento de projetos utilizada para organização de tarefas, definição de sprints e acompanhamento do progresso da equipe [Atlassian 2023]. Foi escolhida por oferecer diversas funcionalidades, sendo particularmente útil para a criação e rastreamento de tarefas ao longo do desenvolvimento.

3.6. Proposta do jogo

O projeto se chama ”Sobreviva”e se trata de um jogo eletrônico do gênero de *survival horror* em primeira pessoa. A história se passa dentro de uma mansão antiga e seus arredores. O jogador, sem nome, acorda preso em uma cadeira e é liberado, sem saber o por que e nem para onde deve ir, ele segue numa jornada para descobrir o que está acontecendo e como sair vivo daquele lugar. A maior parte da história é contado pela própria construção do cenário e algumas notas escritas espalhada pelo mapa. Por isso, o *level design* é bastante importante neste projeto. Ao decorrer da jogatina, o jogador encontrar armas, munições, itens e outros itens que irão auxiliar a enfrentar os desafios que se caracterizam pelos inimigos e enigmas em que o jogador deve descobrir como resolver. Na Figura 2, é possível observar o estilo visual do jogo, a perspectiva em primeira pessoa do jogador segurando uma pistola desenhada em sprite e um HUD minimalista, que destaca a quantidade de munição no pente e a vida do personagem. Ao fundo, o cenário retrata um trecho interno da mansão, combinando elementos 3D e 2D.

Figura 2. Detalhes da composição interna de uma mansão e jogador em primeira pessoa.

A composição do cenário desempenha um papel fundamental na imersão do jogo. Na Figura 3, é apresentado um trecho externo do mapa, com um celeiro ao fundo e uma plantação de milho. A direção de arte foi trabalhada para que os cenários parecessem desenhos animados, o jogo de luz e sombra tem função específica para criar uma atmosfera assustadora, enquanto a iluminação no celeiro destaca o local, orientando o jogador.

Figura 3. Composição de um mapa externo com um celeiro de fundo e uma névoa rasteira.

3.7. Jogabilidade: Sobrevivência em Primeira Pessoa

O jogo adota a perspectiva em primeira pessoa, com mecânicas de movimento, como andar, correr e se abaixar, além de interação com objetos, coleta de itens e combate com armas (faca, pistola e shotgun). O sistema de combate prioriza a precisão, com a mira centralizada ao parar de se mover. A maioria dos inimigos usam ataques a curta distância, o jogador precisa evitar ser encurralado e sempre tentar manter uma certa distância dos inimigos.

3.8. Inimigos: Ameaças na Mansão

O jogo apresenta inimigos com comportamentos distintos, desde criaturas comuns até chefes desafiadores que têm o objetivo de deter o jogador. Apesar dos comportamentos variados, a inteligência artificial dos inimigos é baseada em zonas de influência. Eles atacarão o jogador se este estiver dentro da zona e for visto pelos inimigos.

3.9. Inventário e sala segura

O jogador tem à disposição um inventário com 6 *slots* inicialmente, que podem ser expandidos ao encontrar mochilas, que aumentam em 3 slots, ou pochetes, que aumentam em 1 slot. Alguns itens, como armas, ocupam um slot inteiro, enquanto outros podem ser agrupados, como a munição de pistola, que pode ser agrupada em até 30 cápsulas por slot. A inspiração para o inventário veio do layout apresentado em Resident Evil 7, porém adotando um estilo artístico levemente diferente, para combinar com o visual do jogo.

O jogo também terá uma maleta com espaço infinito. O jogador pode depositar nela itens que possam ser usados futuramente, mas que não precisará levar consigo, deixando-os armazenados lá. Essas maletas serão encontradas nas salas seguras, que não podem ser invadidas por inimigos. Nessas salas, o jogador pode salvar o jogo e reorganizar seu inventário para continuar sua jornada.

3.10. Menus

O jogo possui um menu funcional para iniciar a partida, carregar uma salva e acessar opções para configurar gráficos, como texturas, sombras e outros atributos visuais. Dentro do jogo, é possível pausar para acessar as configurações ou retornar ao menu principal.

Uma partida pode ser salva exclusivamente nas salas seguras, não havendo a opção de salvamento automático. Caso o jogador morra, ele deverá voltar ao ultimo ponto salvo.

3.11. Estilo Artístico: A Fusão do 2D e 3D

O jogo adota um estilo visual que combina elementos 2D e 3D, buscando uma estética de animação 2D com iluminação e ambientes em 3D. As animações das mãos, armas e inimigos são em 2D, enquanto o mapa é construído em 3D com texturas 2D, criando um visual único. A Figura 4 ilustra o carrinho do jogo com texturas em 2D.

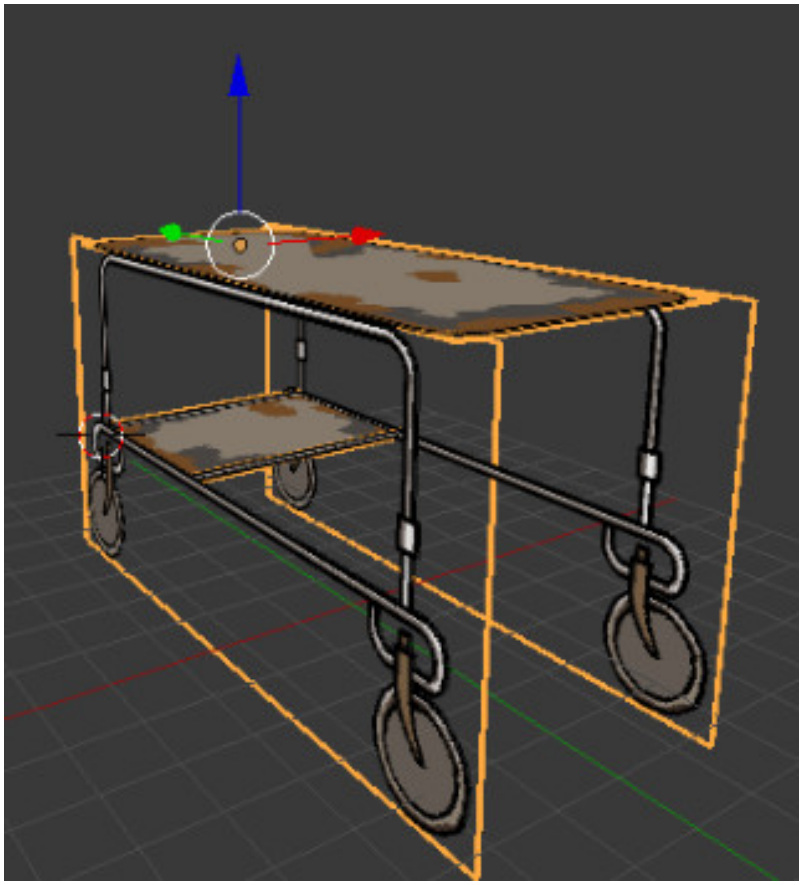

Figura 4. Carrinho de serviço enferrujado modelado apenas com planos com texturas feitas em 2D.

Na Figura 5, é ilustrada a combinação de elementos 2D, que consistem em texturas aplicadas a planos com um script que mantém a face sempre voltada para o jogador, juntamente com modelos 3D texturizados em 2D. Essa abordagem resultou em cenários detalhados, elegantes e distintos da proposta da maioria dos jogos disponíveis no mercado.

Figura 5. A composição do cenário é formado por assets 2Ds e 3Ds.

O método de criação das animações do jogador e dos inimigos combina modela-

gem 3D e desenho frame a frame. Inicialmente, foram criados modelos 3D das armas, das mãos do jogador e dos inimigos. Eles foram animados em 3D e, posteriormente, renderizados frame a frame. Cada frame foi feito com um desenho sobre a referência 3D, estilizado para parecer uma animação 2D, utilizando cores sólidas e três tons de cores para representar zonas de sombra e iluminação mais intensa. A Figura 6 ilustra esse processo.

Figura 6. Da esquerda para a direita: Pistola modelada e animada em 3D, pistola renderizada em frame chave, desenho feito no gimp usando a renderizacao como referência.

4. Resultados e Discussão

Nesta seção, discutiremos a implementação das mecânicas do jogo, os desafios enfrentados e as soluções adotadas, abordando o desenvolvimento do jogador, o sistema de inventário, a criação de mapas e cenários, a inteligência artificial dos inimigos e as técnicas de otimização aplicadas.

4.1. Visão Geral do Jogo

O projeto resultou em um jogo desenvolvido na engine Unity, complementado por softwares de modelagem 3D, edição de imagens e áudio. O objetivo foi criar uma experiência de survival horror, explorando desafios técnicos e conceituais do desenvolvimento de jogos.

4.2. Experiências e Desafios no Desenvolvimento

Nesta seção, serão discutidas as implementações das mecânicas presentes no jogo, seus desafios e os métodos empregados. O desenvolvimento começa com o jogador, que é a base da gameplay, seguido pelo inventário, uma parte fundamental e complexa. Em seguida, discutiremos a abordagem adotada para a composição de mapas, cenários e elementos visuais do jogo. Por fim, abordaremos os inimigos e o sistema de otimização adotado.

4.2.1. Desenvolvimento do Jogador e Mecânicas

O jogador foi construído em primeira pessoa. É possível andar para todas as direções, correr apenas para frente e se abaixar. Embora o pulo tenha sido implementado, ele não foi utilizado no jogo. A criação de escadas ou rampas foi desafiadora. As escadas eram apenas visuais, funcionando com uma rampa transparente com colisão. No entanto, o jogador acabou deslizando para baixo se ficasse parado. Foram necessários alguns ajustes na física para corrigir o problema.

Além disso, a cabeça do jogador foi posicionada ao ajustar a câmera e prender o corpo do jogador. A câmera recebeu o script ”mouselook”, que permite a rotação do eixo da câmera pelo mouse. Esse script possibilita ajustar os ângulos e a velocidade de rotação da câmera. Todas as animações de mãos, seja com armas, itens de cura ou dano, foram feitas quadro a quadro e posicionadas à frente da câmera, de modo que não transbordam nas laterais da tela. As mãos também foram ajustadas em um layout que evita que ”entrem”nas paredes e objetos quando o jogador se aproxima. Na Figura 7, é possível ver o efeito de desfoque ao recarregar a arma. Já na Figura 8, o jogador realiza a animação de cura com uma seringa, que restaura toda a saúde e adiciona uma caveira extra, simbolizando a vida do personagem.

Figura 7. Efeito de desfoque ao mirar.

Figura 8. Animação de cura.

4.2.2. Sistema de Inventário e Gerenciamento de Itens

O sistema de inventário, inspirado em Resident Evil 7, foi um dos maiores desafios do projeto. Inicialmente, o sistema possuía 6 espaços, que podiam ser expandidos ao encontrar pochetes, e esses espaços eram preenchidos com itens, como armas, itens de cura e

itens-chave. A Figura 9 ilustra o inventário do jogo com os itens. Já a Figura 10, demonstra a manipulação de itens dentro do inventário: ao selecionar um dos itens (seringa), abre-se um menu com algumas opções, que podem variar de item para item. No caso da imagem, o jogador pode usar, examinar ou descartar o item.

Figura 9. Menu de inventário com itens de cura, munição e armas.

Figura 10. Os itens podem ser manipulados através de um menu flutuante. Na imagem, a seringa pode ser usada, examinada ou descartada.

Ao clicar em examinar, por exemplo, o item examinado entra em destaque na tela, com um texto descritivo sobre sua funcionalidade ou aparência. A Figura 11 demonstra o jogador examinando a shotgun.

Ao recarregar uma arma, o jogo precisava verificar se havia munição no inventário para continuar o processo. No caso dos itens de cura, o jogador precisava acessar o inventário para usá-los. A manipulação visual do inventário foi um dos maiores desafios, além da lógica de administração de recursos por trás dos códigos. A interface de usuário consistia em vários quadros representando os slots. O jogador poderia focar em um slot e

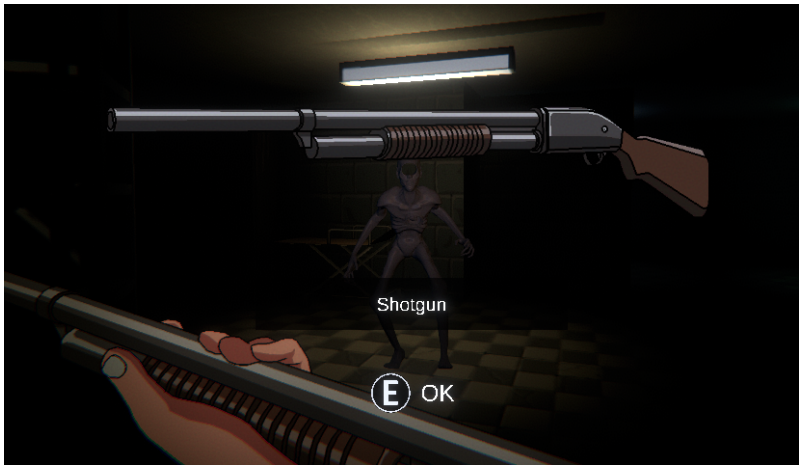

Figura 11. Itens podem ser analisados, contendo textos detalhados sobre seu uso. Na imagem, apresenta-se o rascunho dessa funcionalidade.

abrir outros menus para manipular o item, o que exigia a implementação de várias regras para definir em qual ”seção”do inventário o jogador estava, garantindo a interação correta com os elementos.

4.2.3. Criação de Mapas e Ambientação

O mapa foi projetado com um design de ”backtracking”, incentivando a exploração e a imersão do jogador. A ambientação, inspirada em mansões abandonadas, foi construída com mapas fechados, ideais para combates de curta distância. A criação dos cenários envolveu a combinação de elementos 2D e 3D, com especial atenção à iluminação para criar uma atmosfera de terror e mistério. A Figura 12 ilustra os detalhes externos do cenário da mansão.

Figura 12. Cenário externo da mansão.

Ao começar o jogo, algumas portas e caminhos estão bloqueados, enquanto outros estarão liberados. Conforme o jogador avança, os caminhos serão desbloqueados e as

portas abertas. O design do mapa também foi pensado para garantir combates intensos e curtos, exigindo do jogador rapidez ao se deparar com inimigos.

O mapa foi inspirado em mansões abandonadas e seus arredores. Diversos conceitos e referências de assets foram pesquisados no Pinterest, sendo que as ideias eram desenhadas no papel, escaneadas e, em seguida, desenhadas digitalmente no GIMP. Alguns objetos eram 2D e ficavam sempre voltados para frente do jogador, utilizando uma técnica popular em jogos antigos como Doom. Já outros objetos, como mesas e cadeiras, foram modelados em 3D, mas com o uso de objetos simples, como planos, para compor a estrutura geral do modelo. Isso contribuiu para a criação de um visual coeso.

A iluminação também foi uma peça-chave do projeto, posicionada em locais estratégicos para iluminar apenas os elementos de interesse, deixando o restante do mapa no escuro ou com pouca iluminação. O objetivo era criar um cenário assustador e estranho. A Figura 13 demonstra o trabalho de iluminação. Como o jogo se passa em ambientes pouco iluminados, os efeitos de luz e sombra foram elaborados de maneira a serem úteis e elegantes, destacando ainda mais os detalhes artísticos nos cenários. Para o efeito de traçado de luz na janela, foi usado um *pack* chamado aura, disponível de forma gratuita na loja da Unity.

Figura 13. Composição de mapa interno com a luz apresentando o efeito de raio de luz entrando pela janela, iluminando uma parte do mapa.

4.2.4. Inteligência Artificial e Comportamento dos Inimigos

A inteligência artificial dos inimigos foi projetada para ser simples, baseada em zonas de influência e comportamentos aleatórios, a fim de simular imprevisibilidade. Quando o inimigo se aproxima de uma determinada distância, a animação de ataque é disparada e, caso o jogador esteja dentro da zona de ataque, o dano é aplicado. Além disso, foram implementadas probabilidades para que o inimigo altere seu percurso ao atacar, tornando seu comportamento menos previsível.

Inicialmente, o jogo teria nove inimigos comuns e cinco chefões. No entanto, até o momento, apenas um inimigo foi desenvolvido. O dano causado ao jogador varia conforme a área atingida, e ataques à cabeça possuem uma pequena chance de causar morte instantânea. Da mesma forma, o dano recebido pelo inimigo também é influenciado pela região do impacto.

Visualmente, o inimigo foi criado a partir de um modelo 3D, animado por meio de rigging no Blender e renderizado frame a frame. A Figura 14 ilustra o combate entre o jogador e o inimigo dentro do cenário do jogo com o inimigo já desenhado frame a frame.

Figura 14. Jogador combatendo inimigos comuns, com inimigos desenhados e animados no GIMP.

Na Figura 15, para manter a precisão no disparo, o jogador precisa permanecer parado, sem incluir a rotação do corpo, para que a mira se feche em um único ponto e o disparo seja mais concentrado, causando mais dano ao inimigo. Quando o jogador se move, a mira se expande, indicando que a precisão foi perdida, como mostrado na Figura 16.

Figura 15. Combate contra inimigos. A mira da arma fica fechada, demonstrando precisão enquanto o jogador permanece parado por um tempo.

Figura 16. Quando o jogador anda ou atira, ele perde precisão, fazendo com que a mira se abra, indicando que o tiro pode não ser preciso.

4.2.5. Interação com o cenário

Durante o jogo, o jogador poderá interagir com alguns itens, como portas e gavetas, para encontrar itens, recursos ou liberar passagens. A interação é baseada na distância do jogador em relação ao objeto. Ao se aproximar, o objeto exibe um ícone de interação. No entanto, isso causou um problema para itens interagíveis próximos.

Para resolver esse problema, foi adotado, além do sensor de aproximação, a visão do jogador. Objetos mais centralizados em relação ao ponto de vista do jogador serão considerados prioridade na interação. Na Figura 17, o jogador abriu a gaveta e encontrou um item de cura. As duas interações estão próximas, mas a câmera está posicionada de forma que o item fica mais centralizado, dando prioridade a essa interação.

Figura 17. Jogador interagindo com uma gaveta e encontrando um item dentro dela.

O movimento das portas e gavetas foi feito manipulando sua rotação ou posição, evitando a criação de animações. na figura 18 mostra o jogador abrindo a porta de um armário, não tendo itens lá dentro.

Figura 18. Jogador interagindo com uma porta de um armário.

4.3. Sons

Trilhas e efeitos sonoros são um dos principais desafios para equipes indie (discutido na subseção 2.2). Isso ocorre porque é muito difícil encontrar profissionais qualificados para produzir esses efeitos, e amadores frequentemente têm dificuldades em alcançar resultados satisfatórios. Os sons são elementos sensoriais cruciais de um jogo. Além disso, é necessário um investimento considerável em equipamentos caros para capturar áudio de maneira convincente e sem ruídos. Muitas equipes indie acabam dependendo de efeitos sonoros gratuitos e sem direitos autorais, que foi o caso deste projeto.

4.3.1. Sons dos personagens

Sons produzidos diretamente ou indiretamente pelos personagens, como o jogador ou inimigos. Exemplos incluem passos, tiros, recarga de armas, ataques, danos, itens e outros efeitos sonoros gerados pelos personagens. Esses sons são únicos, característicos e mais difíceis de produzir ou encontrar sem infringir direitos autorais. Além disso, são mais notados pelos jogadores devido à sua importância no jogo. Normalmente, esses sons são sincronizados com animações específicas, tornando-os ainda mais impactantes.

4.3.2. Sons ambiente com objetos fontes

Sons emitidos por objetos no cenário, como lâmpadas, motores e televisores ligados. Esses sons têm a particularidade de apresentar um efeito 3D: quanto mais próximo o jogador está do objeto fonte, mais alto é o som, criando uma sensação de realismo no ambiente. Na figura 19, uma fonte de som foi associada ao modelo 3D de uma lampada. O som é 3D, ou seja, quando o jogador se aproxima, o som fica mais auto, quando se afasta, o som fica mais baixo.

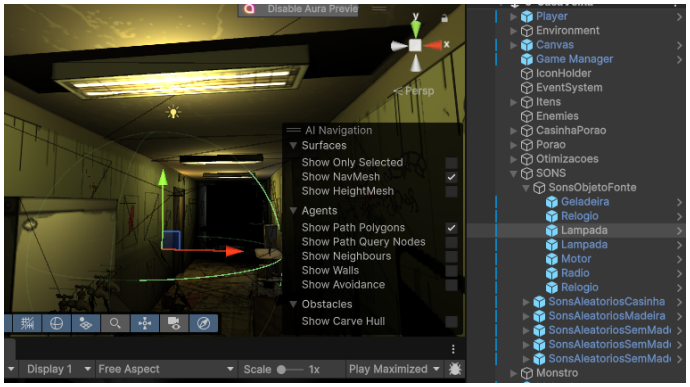

Figura 19. Som da vibração da luz sendo inserido em um asset de lampada no cenário.

4.3.3. Sons ambiente aleatórios

Sons produzidos de maneira aleatória pelo cenário para aumentar a imersão do jogador, como madeira rangendo, objetos caindo e barulho de mato. Vários pontos vazios são distribuídos pelo mapa, e um script escolhe aleatoriamente um desses pontos para reproduzir um dos sons de ambiente disponíveis. Isso cria a sensação de que os sons estão vindo de diferentes direções e distâncias, enriquecendo a experiência sonora. Na Figura 20, cada quadrado apresenta uma zona com vários pontos de sons. De maneira aleatória, é escolhido um desses pontos para reproduzir algum tipo de som.

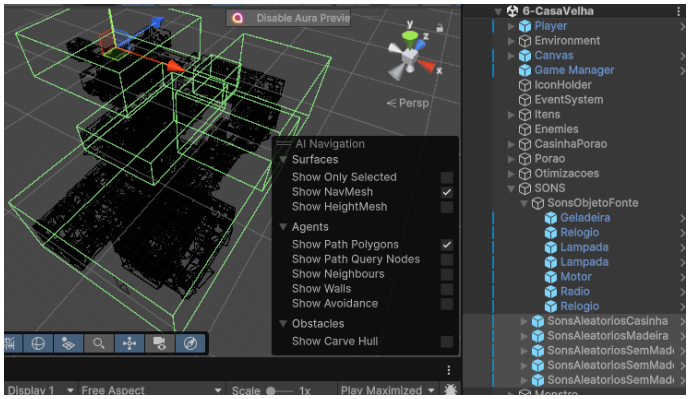

Figura 20. Jogador interagindo com uma porta de um armário

4.3.4. Sons de interação

Sons produzidos quando o jogador interage com objetos no jogo. Esses sons podem ser realistas, como os gerados ao interagir com elementos físicos do cenário (por exemplo, portas ou gavetas), ou fictícios, como os efeitos sonoros associados à interação com o inventário ou sistemas de interface. Esses sons ajudam a reforçar a sensação de interação e resposta do ambiente ao jogador.

4.3.5. Trilhas sonoras

Músicas instrumentais que tocam em momentos-chave do jogo, como durante batalhas contra chefes, para criar o clima adequado. No início do desenvolvimento, o projeto não possuía faixas originais. Foram utilizadas algumas trilhas do jogo Resident Evil 7 como testes, com a promessa de que, posteriormente, faixas originais seriam criadas para o jogo. Essas trilhas desempenham um papel crucial na definição da atmosfera e na intensificação da emoção durante momentos importantes do jogo.

4.4. Otimização do Desempenho do Jogo

A otimização de jogos eletrônicos é uma das partes fundamentais neste tipo de projeto, mas muitas vezes negligenciadas em equipes independentes, o que pode resultar em jogos pesados, mesmo quando os visuais são simples. Por isso, desde o início, a otimização do projeto foi uma das prioridades.

Técnicas como o Mipmapping, que já vêm automaticamente na Unity, foram aplicadas. Essa técnica ajusta a resolução das texturas dependendo da distância: as mais distantes são exibidas com resoluções mais baixas, enquanto as mais próximas são renderizadas com mais detalhes. Além disso, removemos as sombras de objetos que não precisavam ser renderizados, otimizando ainda mais o desempenho.

Outra técnica muito utilizada na indústria de jogos eletrônicos é o LOD (Level of Detail). No entanto, essa técnica não se encaixou bem no projeto deste artigo por dois motivos: os modelos, mesmo os 3D, já são simplificados ao máximo, e o cenário é composto principalmente por corredores pequenos. Assim, o efeito ”LOD Popping”(mudança repentina de detalhes) seria muito mais perceptível. A Figura 21 mostra um sensor de otimização. Quando o jogador passa por ele, é detectado qual face foi interagida primeiro, a fim de determinar a direção em que o jogador está indo. Com isso, um trecho do mapa é carregado enquanto outro é removido, como ilustrado nas Figuras 22 e 23.

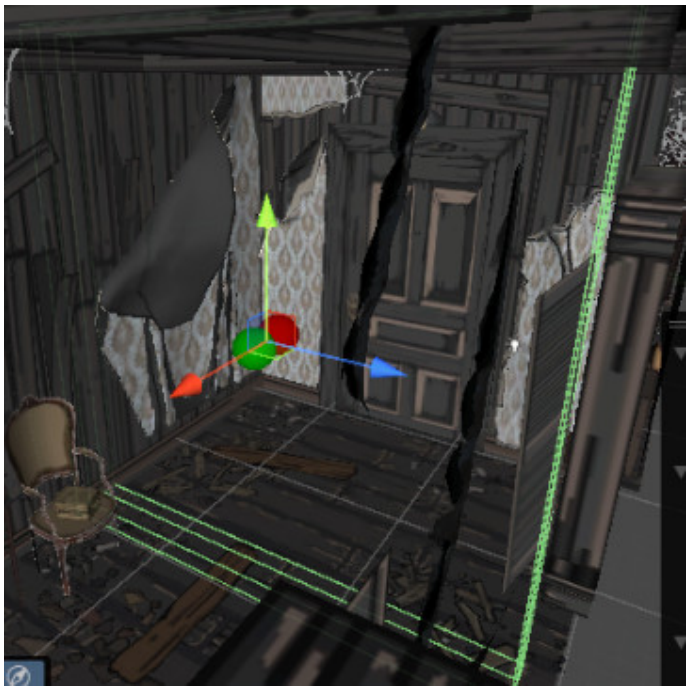

Figura 21. Sistema de otimização. No lado da esfera vermelha, os mapas selecionados serão removidos do mapa, quando o jogador passar pela esfera verde, os mapas serão inseridos novamente no mapa.

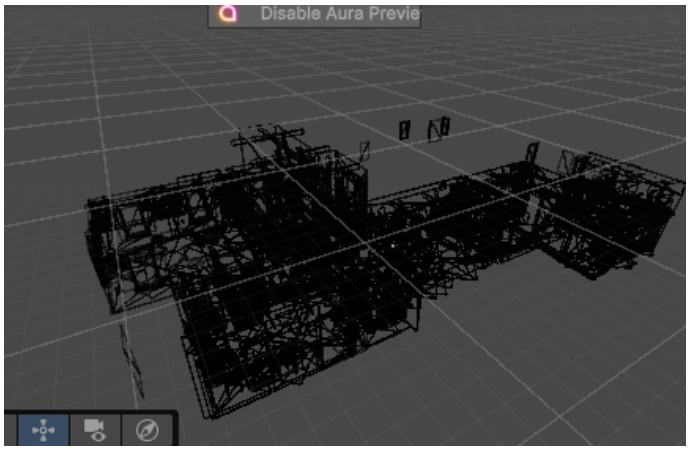

Figura 22. Quando o jogador está em uma parte do mapa, o cenário é renderizado.

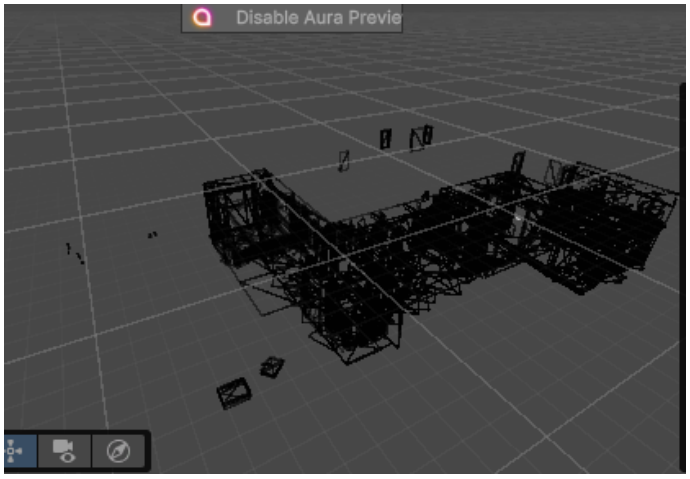

Figura 23. Ao passar pelo sensor, um trecho do mapa é removido e outro inserido.

Para contornar esse problema, foi criado um script que funciona da seguinte maneira: aproveitando que os cenários são fechados, foram colocados sensores em locais estratégicos do mapa. Quando o jogador se aproxima de um desses sensores, apenas os cenários mais próximos são renderizados, e o restante do mapa é removido. À medida que o jogador se move, diferentes trechos do mapa são renderizados e ocultados.

Para evitar que a remoção de cenários fosse perceptível, algumas portas foram configuradas para se fecharem automaticamente quando o jogador se distanciava delas, ocultando o cenário removido de forma discreta.

O resultado foi bastante satisfatório, proporcionando uma experiência de jogo mais otimizada sem que o jogador notasse mudanças abruptas no ambiente.

5. Conclusão

O desenvolvimento do projeto, desde a concepção até o estágio atual, proporcionou uma experiência extremamente enriquecedora. A criação de jogos eletrônicos envolve desafios técnicos, como o domínio de linguagens de programação, e artísticos, como modelagem, design e até edição de vídeos e áudios. Além disso, o trabalho em equipe desenvolve habilidades pessoais importantes na gestão e organização de projetos. Cada etapa do processo exigiu soluções criativas, que demandaram pesquisa e expansão de conhecimento em diversas áreas, instigando a busca por soluções únicas.

5.1. Contribuições

O desenvolvimento do jogo Sobreviva proporcionou uma experiência enriquecedora, desde sua concepção até o estágio atual do projeto. A criação de jogos eletrônicos envolve desafios multidisciplinares, exigindo conhecimentos técnicos em programação e habilidades artísticas em modelagem, design e edição audiovisual. Além disso, o trabalho em equipe aprimorou as habilidades de gerenciamento e organização de projetos. Cada etapa exigiu pesquisa, aprendizado e soluções criativas.

5.2. Limitações e Trabalhos Futuros

Apesar da crescente disponibilidade de ferramentas gratuitas e do suporte da comunidade, o desenvolvimento de jogos eletrônicos ainda enfrenta limitações, especialmente para

equipes independentes.

- **Limitações Financeiras**: A necessidade de conciliar o desenvolvimento com outras atividades e a dificuldade de adquirir hardware adequado são desafios comuns.
- **Desafios do Mercado Brasileiro**: O alto custo de hardware e a falta de incentivos específicos dificultam o desenvolvimento de jogos eletrônicos no Brasil.

O projeto "Sobreviva", em desenvolvimento há dois anos, ainda está em progresso. Os planos futuros incluem a criação de uma fase jogável completa para testes com o público, refinamento das mecânicas e aprimoramento dos aspectos do jogo.

Referências

Associação Brasileira das Desenvolvedoras de Jogos Digitais (Abragames) (2023). O jogo muito além dos games. Acessado em: 10 mar. 2025.

Atlassian (2023). Jira Software - Issue Project Tracking for Software Teams. Acessado em: 10 mar. 2025.

Atlântico (2023). O papel da indústria de games no avanço das tecnologias 3D e Digital Twins. Acessado em: 10 mar. 2025.

Audacity Team (2023). Audacity - Free, open source, cross-platform audio software. Acessado em: 10 mar. 2025.

Barbosa, B. F. (2015). As características da implementação de um jogo survivor horror. Trabalho de Conclusão de Curso, Centro Paula Souza. Acessado em: 17 mar. 2025.

Be Bold Comunicação (2023). Indústria dos games fatura mais que a do cinema e da música – e sim, é para os adultos também. Acessado em: 10 mar. 2025.

Blender Foundation (2023). Blender - The Free and Open Source 3D Creation Suite. Acessado em: 10 mar. 2025.

Canaltech (2021). Mercado de games agora vale mais que indústrias de música e cinema juntas. Acessado em: 10 mar. 2025.

do artigo, A. (2013). Fatores de sucesso para a indústria de jogos digitais. In *Anais do SBGames 2013*. Acessado em: 10 mar. 2025.

GIMP Team (2023). GIMP - GNU Image Manipulation Program. Acessado em: 10 mar. 2025.

GitHub (2023). GitHub - Where the world builds software. Acessado em: 10 mar. 2025.

Instituto Claro (2023). Uso de simuladores permite aprendizado prático e baixa custos em muitas áreas de formação profissional. Acessado em: 10 mar. 2025.

Jornal da USP (2023). A expansão do mercado de games brasileiro se deve a mudanças no modo tradicional do trabalho. Acessado em: 10 mar. 2025.

José Hícaro Hellano (2019). O Uso da Estratégia Gameficação na Educação Médica. *Revista Brasileira de Educação Médica*, 43(1):58–65. Acessado em: 10 mar. 2025.

Maria Carolina Ortiz Whitaker (2021). Jogos eletrônicos na atenção à saúde de crianças e adolescentes: revisão integrativa. *Acta Paulista de Enfermagem*, 34(eAPE02731). Acessado em: 10 mar. 2025.

Microsoft (2023). Visual Studio Code - Code Editing. Redefined. Acessado em: 10 mar. 2025.

Ministério da Ciência, Tecnologia e Inovação (2023). Mercado potencial de games no brasil é tema de audiência. Acessado em: 10 mar. 2025.

NetCoders (2023). Gamedev: Motores de Jogo – Visão Geral e Comparação. Acessado em: 10 mar. 2025.

Portal Correio (2025). O crescimento do mobile gaming no Brasil: tendências e desafios. Acessado em: 10 mar. 2025.

Ribeiro, A. C. M. (2015). A indústria de vídeo games e seu impacto econômico. Trabalho de Conclusão de Curso, Universidade Federal de Mato Grosso. Acessado em: 10 mar. 2025.

Salutes, B. (2023). O que é motor gráfico? Acessado em: 10 mar. 2025.

TechTudo (2023). Brasil é o maior mercado gamer da américa latina e top 10 mundial em 2023. Acessado em: 10 mar. 2025.

Unity Technologies (2023). Unity - Plataforma de Desenvolvimento de Jogos. Acessado em: 10 mar. 2025.

Veja (2023). Mudou de fase: mercado de games já fatura mais que o de cinema. Acessado em: 10 mar. 2025.