Design de níveis num videojogo multijogador

(The abyss)

um projeto realizado por

João Pedro Pinto Ribeiro

supervisionado por

Prof. Cláudia Sofia Borlido de Freitas

Professor Assistente, ISMAI

Prof. Agostinho Gil Teixeira Lopes

Professor Auxiliar, ISMAI

Mestrado em Tecnologias de Informação, Comunicação e Multimédia no Universidade da Maia - ISMAI (ISMAI)

10 de fevereiro de 2023

Resumo

Videojogos, do ponto de vista académico, têm sido estudados em relação às alterações que provocam nas pessoas, em particularmente nos jovens que são a grande maioria que utilizam os mesmos. Os videojogos provaram também ser uma ótima fonte de aprendizagem e que, ao contrário de crenças antigas, podem tornar o utilizador um pouco mais reativo e mais conectado com outros indivíduos.

No âmbito deste trabalho, pretende-se descrever os que são considerados videojogos, tanto do ponto de vista académico, como do ponto de vista de lazer, como são desenvolvidos e o âmbito para que são criados. É revisto o planeamento para desenvolver um videojogo, com recurso à tecnologia Unity Engine, com especial foco em design de níveis num jogo multijogador e otimização de níveis. Serão revistas técnicas de design de níveis, mecânica, jogabilidade e diversas técnicas de otimização. No decorrer deste trabalho, irá também ser mencionado outras ferramentas disponíveis para a realização de um videojogo, tais como diferentes motores de jogos disponíveis no mercado e ferramentas já existentes, como também diferentes abordagens para a criação deste. Durante o desenvolvimento é também referido os processos de otimização de níveis utilizados, como este impactaram o mesmo e o porquê de serem importantes.

Por último, será apresentado o resultado final, com todo o conteúdo e aprendizagem sucedida, com especial foco nas grandes vantagens que um projeto com esta magnitude proporciona, tanto a nível académico como a nível profissional e uma área de debate e reflexão sobre questões colocadas durante a fase de testes do projeto.

Palavras-chave: Level design. Multijogador. Inteligência artificial. Otimização de videojogos.

Abstract

From an academic point of view, video games have been studied in relation to the changes they cause in people, particularly in young people, who are the vast majority who use them. Video games have also proved to be a great source of learning and that, contrary to old beliefs, they can make the user a little more reactive and more connected with other individuals.

Within the scope of this paper, it is intended to describe what are considered video games, both from an academic and leisure point of view, how they are developed and the scope for which they are created. The planning for developing a video game, using Unity Engine technology, is reviewed, with a special focus on level design in a multiplayer game and level optimisation. Level design techniques, mechanics, gameplay and various optimisation techniques will be reviewed. During the course of this work, other tools available for making a videogame will also be mentioned, such as different game engines available on the market and existing tools, as well as different approaches to creating one. During the development it will also be mentioned the level optimization processes used, how they impacted it and why they are important.

Finally, the final result will be presented, with all the content and learning achieved, with special focus on the great advantages that a project of this magnitude provides, both academically and professionally, and an area for debate and reflection on issues raised during the testing phase of the project.

Keywords: Level design. Multiplayer. Artificial intelligence. Game optimization.

Agradecimentos

Gostaria de agradecer aos meus pais e amigos, por todo o apoio durante o desenvolvimento desta dissertação, sem a ajuda destes, isto não seria possível.

Gostaria também de agradecer aos meus orientadores, professora Cláudia Sofia Borlido de Freitas e professor Agostinho Gil Teixeira Lopes, por todo o auxilio prestado e extrema dedicação, durante toda a etapa desta dissertação.

Conteúdo

Lista de Figuras

Siglas

AI Artificial Intelligence

Co-op Cooperative Video Game

CPU Central Processing Unit

FPS First-Person Shooter

GPU Graphics Processing Unit

HDRP High Definition Render Pipeline

ISMAI Universidade da Maia - ISMAI

MMO Massively Multiplayer Online Game

MOBA Multiplayer Online Battle Arena

MTICM Mestrado em Tecnologias de Informação Comunicação e Multimédia

NPC Non-Player Character

PVE Player Versus Environment

PVP Player Versus Player

RAM Random-Access Memory

RPG Role-Playing Game

TPS Third-Person Shooter

URP Universal Render Pipeline

Glossário

AI Inteligência artificial

Asset Objeto utilizado no videojogo

Bug Erro no sistema que provoca um comportamento inesperado

Co-op Termo dado a um videojogo cooperativo

CPU Unidade de processamento central

Cross-Platform Opção que existe quando a aplicação é suportada entre várias plataformas, incluindo consolas ou telemóveis

Feedback Informação sobre a reação de um produto, normalmente utilizado para melhoramento

FPS Estilo de videojogo em que o jogador assume o controlo da personagem de um ponto de vista primeira pessoa

Game Engine Motor de desenvolvimento onde é criado o videojogo

Gameplay Jogabilidade do videojogo

GPU Unidade de processamento gráfica

HDRP Pipeline de renderização de alta definição do Unity

HIGH-POLY Estilo de objeto de modelação com muitos polígonos

Indie Equipas de produção independentes

Input Ato de inserir algo num programa

Level Design Cargo encarregue de desenvolver e produzir o mapa do videojogo, tanto como as interações, mecânicas, jogabilidade e otimizações da cena

Loading Screen Ecrã de espera apresentado quando o jogador altera de cena, normalmente utilizado quando é necessário carregar materiais para uma cena nova

LOW-POLY Estilo de objeto de modelação com poucos polígonos

MMO Termo dado a um videojogo com um numero elevado de utilizadores em simultâneo

MOBA Estilo de videojogo que proporciona uma arena aos jogadores para lutarem entre si ou contra inimigos

Networking Aspeto relacionado com o ambiente de rede do videojogo

NPC Termo dado a uma personagem ou objeto não jogável

PVE Termo dado a um videojogo com o foco do jogador contra o computador

PVP Termo dado a um videojogo com o foco do jogador contra outros jogadores

RAM Memória de acesso aleatório

RPG Estilo de videojogo em que o jogador assume o papel de uma personagem e joga com as narrativas da mesma

Scripting Criação de código que não necessita de ser compilado para correr

Sprites Termo dado a imagens 2D quando integradas numa cena de um videojogo

STYLIZED Estilo de arte de um videojogo, normalmente não realista

TPS Estilo de videojogo em que o jogador assume o controlo da personagem em terceira pessoa, ou seja, neste modo a personagem é totalmente visível ao jogador

URP Pipeline de renderização universal do Unity

Capítulo 1

Introdução

O termo videojogo tem vindo a surgir cada vez mais no mundo académico, tal que muitos académicos possuem diferentes definições para descrever o que é um videojogo. Mas então, o que é um videojogo?

"Contudo, mesmo que pareça óbvio, videojogos são, antes de tudo, jogos."[1]

Mark Wolf defende que para definir videojogos, deve-se ter em conta diferentes aspetos, tais como tecnologia, arte e natureza da experiência, mas que na sua essência, videojogos são a junção do termo "vídeo" com o termo "jogo". Wolf acrescenta também que elementos comuns encontrados no termo "jogo", podem ajudar a definir o termo videojogo, tais como regras, determinando o que pode ou não ser feito, conflitos, contra um oponente ou circunstâncias, habilidades do jogador, tal como estratégia ou sorte e algum tipo de recompensa[2].

A indústria de videojogos tem vindo a crescer bastante, ultrapassando em 2020 a indústria de cinema e de desporto norte-americano, faturando €180 biliões, sendo que os analistas prevêem que este crescimento continue após o lançamento das novas consolas (Playstation 5/ Xbox Series X)[3].

Num estudo realizado pela autora Dina Bassiouni, crianças de um espetro de idades entre os 12 e 18 anos, utilizam regularmente videojogos e tem afetado de alguma forma as suas capacidades cognitivas. Bassiouni defende que nas últimas pesquisas, os jovens atuais estão mais avançados do que as gerações passadas, e que ao contrário das crenças antigas, de que os videojogos prejudicavam os encontros cara a cara dos jovens, estes pelo contrário incentivam estes encontros[4].

Também o autor Williams defende que os humanos tendem a utilizar os videojogos para se conectaram uns com os outros e que a tecnologia fornece métodos de melhoramento nestas conexões[5].

Videojogos têm vindo a quebrar barreiras, tanto a nível social como educacional, diz o autor Egenfeldt-Nielsen. O mesmo defende, que após vários estudos realizados com o intuito de verificar o impacto dos videojogos a nível educacional, os resultados foram mistos, variando muito o sentido à qual os videojogos foram utilizados. Com isto, algumas áreas não provaram ter alterações significativas, enquanto outras áreas, tais como as do conhecimento, provaram ser bastante efetivas, no sentido de utilizar videojogos para fins de aprendizagem[6].

Perante estes estudos, percebeu-se desde cedo a autenticidade e o nível de exigência, em desenvolver um projeto, que não só fizesse puxar pelos limites do conhecimento, mas que também permitisse outros indivíduos apreciar e explorar um mundo, que muitas das vezes é completamente novo. Com isto em mente, decidiu-se juntar várias áreas e diferentes métodos de abordagem ao projeto, de forma a utilizar tecnologia recente e inovadora.

Este projeto tem como foco a criação de um videojogo, com suporte multijogador, utilizando a ferramenta Unity. Será detalhado todo o processo desde a conceção dos níveis, otimizações, mecânicas e jogabilidade, com especial foco no jogador. O videojogo será em desenvolvido em estilo Role-Playing Game (RPG), com elementos Third-Person Shooter (TPS) e uma atmosfera imersiva. Será também usado o recurso a inteligência artificial para dar vida aos níveis e aos Non-Player Characters (NPCs).

1.1 Problemática e Motivação

Projeto "The Abyss" surgiu de uma ideia em conjunto, de forma a conciliar conhecimentos e tecnologia recente, para a conceção de um videojogo que ultrapassasse as capacidades atuais dos autores do mesmo, de modo a que pudesse ser utilizado como fonte de aprendizagem e para o âmbito de portefólio profissional. A motivação por detrás deste projeto provêm do fato que este tipo de projetos, não só requerem tamanho nível de conhecimento e prática, como requerem normalmente uma equipa com um budget alto. Com isto, surgindo a oportunidade única de experienciar e aprender com um projeto desta escala, juntaram-se quatro elementos num grupo, dois artistas e dois programadores (Fig. 1.1).

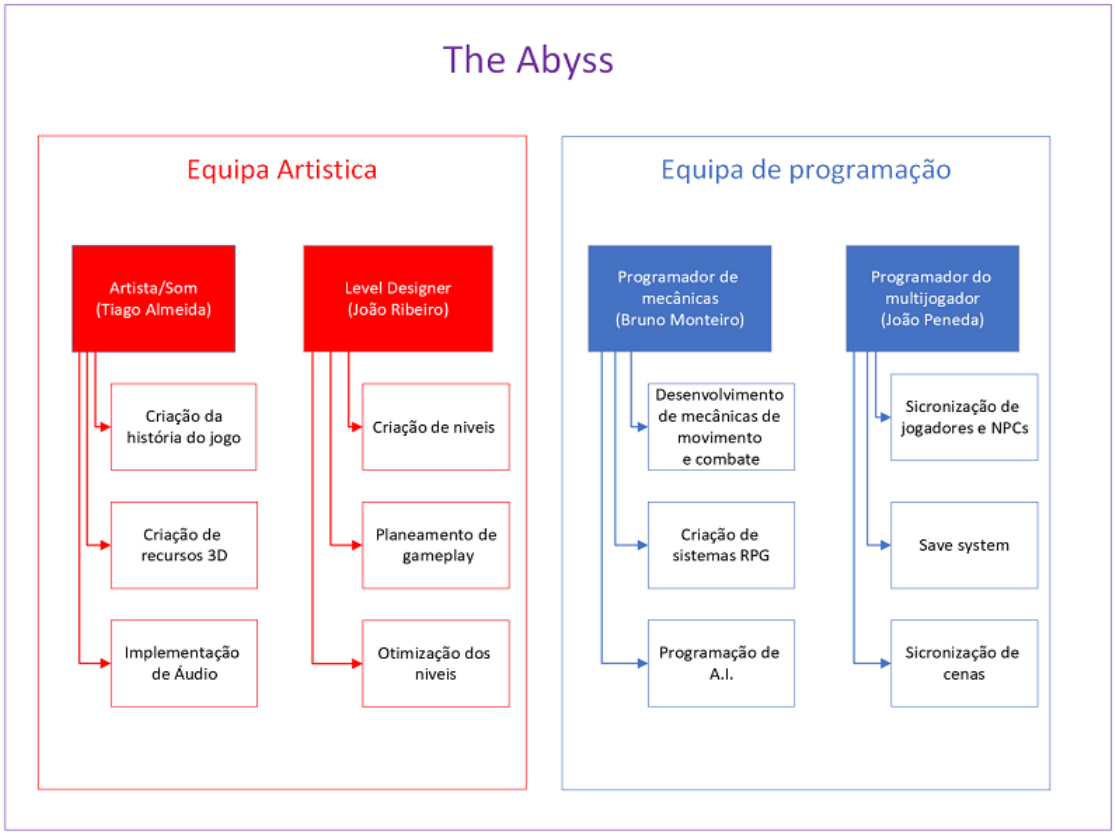


Figura 1.1: Estrutura do grupo de projeto

Uma vez na fase de análise, foi efetuada uma pesquisa extensiva sobre a tecnologia recente existente e o que havia relatado sobre a mesma. Para tal, foram revistos imensos artigos científicos e implementações reais deste tipo de software, para que a base do projeto fosse viável. Após a analise, decidiu-se realizar um videojogo, que combina dois estilos

bastante conhecidos, RPG e TPS, com a adição de multijogador.

O foco principal do projeto é de proporcionar ao jogador uma jogabilidade única, uma atmosfera dinâmica e viva e uma forma de partilhar esta experiência com alguém. Para isto, o projeto foi todo pensado num ambiente multijogador, que embarga os jogadores num ambiente único e uma história aliciante.

Atualmente, uma porção de videojogos desenvolvidos com temas semelhantes, possuem características menos positivas comuns entre eles, desde má otimização, estilo não uniforme, limitações nas funções de multijogador e de jogabilidade. "The Abyss" surge para ilustrar a área do desenvolvimento de videojogos, com especial foco na utilização de diferentes ferramentas utilizadas atualmente pela indústria e construir algo único e dinâmico. Contudo, isto necessita de um inúmero conjunto de regras e metodologias, de forma a tornar o videojogo acessível e bom para o jogador. Um dos problemas comuns deparados durante a pesquisa, foi com o facto de o videojogo ter de funcionar em multijogador e para tal, todo o tipo de ações no mundo têm de ser processadas pelo lado do servidor, de modo a que todos os jogadores consigam ver ações em sincronia, ações estas como ver outro jogador correr ou ouvir sons emitidos por outros jogadores na sessão. Entretanto, a otimização da cena e de tudo o que é renderizado para o jogador, desempenha um papel importantíssimo, pois qualquer erro durante este processo pode comprometer a experiência do jogador, mas como também se corre o risco de o jogador nem conseguir jogar o jogo.

No entanto, o projeto continua a ser um videojogo e uma das maiores problemáticas acaba por ser a qualidade da inteligência artificial. Caso esta seja fraca, a imersividade e consequentemente a diversão do jogador será exponencialmente menor, podendo até mesmo ser um motivo para não jogar o videojogo. Daí a implementação de uma boa inteligência artificial distingue imensos videojogos já existentes e é pela mesma razão que é tão importante para este projeto.

1.2 Objetivos

Objetivo Geral

Definir e melhorar práticas de desenvolvimento de um videojogo, com ênfase no desenvolvimento dos níveis, otimizações de cenas e jogabilidade.

Objetivos Específicos

- Analisar a literatura sobre ferramentas e metodologias de desenvolvimento de videojogos;
- Definir erros comuns analisados e como os resolver;
- Fundir ferramentas profissionais de desenvolvimento de videojogos, de modo a produzir algo único;
- Construir os níveis do videojogo, com ênfase nos desafios, ambiente, jogabilidade, imersão e inteligência artificial;
- Analisar todos os recursos utilizados pela Graphics Processing Unit (GPU), Central Processing Unit (CPU), e Random-Access Memory (RAM) de modo a garantir que recursos não sejam desperdiçados;
- Otimizar os objetos em cena, de modo a garantir performance em diversos sistemas e uma jogabilidade estável;
- Desenvolver uma interface gráfica básica para o jogador.

1.3 Organização de capítulos

No capítulo de introdução é feito uma abordagem ao videojogos, do ponto de vista social e académico. É descrito o problema e os objetivos deste projeto.

Durante o capítulo do estado da arte é feita uma revisão de literatura e pesquisa sobre o estado atual do desenho de níveis, com foco no multijogador. São apresentadas ideias de diferentes autores e algumas implementações atuais. São também revistas técnicas para ultrapassar e lidar com problemas durante o desenvolvimento de níveis, bem como a diferença entre aspetos multijogador.

Para o capítulo de planeamento é referido o método de investigação utilizado no desenvolvimento do projeto, bem como é definida a implementação do mesmo, testes a realizar recursos necessários e o cronograma de gantt.

No capítulo de desenvolvimento é descrito todo o planeamento realizado e todas as etapas de desenvolvimento, divididas por partes, para o videojogo. Contém também um capítulo referente à otimização do mesmo.

O capítulo de testes, descreve o processo de realização dos testes externos e de como estes foram distribuídos. Este capítulo possui também uma área de discussão de resultados dos mesmos.

Por último, é agregado todo o conteúdo para a conclusão do projeto, sendo que o capítulo de desenvolvimento e o capítulo de testes são levados em conta para o desenvolvimento do capítulo de trabalho futuro.

Capítulo 2

Estado da arte

Este capítulo descreve, de um ponto de vista literário, o que existe atualmente para detalhar o que é um videojogo, como este é desenvolvido e pontos críticos no desenvolvimento.

Foi realizada uma pesquisa de literatura sobre ferramentas de desenvolvimento de videojogos, tipos de jogabilidade, impacto da interatividade entre o videojogo e o jogador e a importância de uma boa inteligência artificial. Foram também revistas ferramentas de desenvolvimento de videojogos como Game Engines, ferramentas auxiliares de criação de níveis e estruturas lógicas na conceção de um nível. Enquanto o foco da pesquisa foi nas práticas de conceção de um videojogo do ponto de vista de Level Design, outros pontos fulcrais são também abordados. Só material citado/publicado de livre acesso foi incluído.

2.1 Videojogos

Desde cedo, os humanos jogam jogos, seja com o propósito de lazer ou competição. Com o avanço tecnológico, os videojogos foram evoluindo, desde os jogos tradicionais até ao videojogos atualmente. Embora o termo videojogo adicione o ênfase de virtual ao jogo, os objetivos do mesmo permaneceram intactos. Desde contar uma história até simular uma experiência única, os videojogos abriram oportunidades aos humanos de explorar novos horizontes com os jogos.

2.1.1 Level Design

Level Design em videojogos é referido como o cargo encarregue da conceção do mundo digital, desde os locais de acesso no jogo, tipos de missões, objetivos ou até mesmo mecânicas que envolvem o jogador. John Feil compara o ato de Level Design a pinturas de massa, ou seja, neste tipo de pinturas o artesão tenta criar uma pintura numa folha de papel, colando peças de massa com cola, mas o artesão não faz a massa nem a cola utilizada, contudo ao utilizar estes componentes, ele cria arte[7].

Level Design pode ser dividido entre as seguintes fases:

- **Esboço em papel** - Durante esta fase é feita uma recolha de ideias em papel e é posta em prática os primeiros esboços de cada nível.
- **Construção do terreno** - Com o auxílio dos esboços, é então feita a transição do papel para a game engine, sendo nesta fase que é criado o terreno de jogo, as texturas base e a estrutura lógica do nível, tanto a nível do jogador como a nível da Inteligência artificial, denominada NavMesh.
- **Popular espaços e adicionar estruturas** - Mesmo com o terreno finalizado, este não está completo, tendo de ser populado, geralmente com a criação de cidades, vilas ou até mesmo campos de futebol.
- **Luz e efeitos atmosféricos** - Nesta fase é introduzida a "magia" no jogo, com os diversos efeitos de luz colocados de modo a criar um ambiente atrativo visualmente, mas tal como efeitos como nevoeiro, ambientes de som ou partículas.
- **Respirar vida para os níveis** - Durante esta fase é feita a adição de NPCs, interações com o jogador, desafios, pilares para o diálogo e missões.
- **Polir o nível** - Uma das partes mais importantes no desenvolvimento de um nível é conseguir otimizar o mesmo para funcionar nas melhores opções disponíveis. Para isto todo o nível é submetido a uma análise extensiva, onde é avaliado cada objeto utilizado, incluindo luzes de ambiente e é efetuado um tipo de renderização especial em conjunto com umas funções de otimização, para garantir a fiabilidade do nível.

2.1.2 Game Engines

Game Engine é um termo comum na indústria de videojogos, para descrever uma plataforma de auxílio para a criação de videojogos. Este tipo de plataforma existe para abstrair detalhes como renderização, físicas e Inputs, para que utilizadores consigam se focar nos detalhes essenciais para tornar os seus jogos únicos.

Michael Lewis adiciona também que Game Engines oferecem componentes reutilizáveis que podem ser manipulados para dar vida a um videojogo, tais como modelos de animação, Loading Screens, modelos de colisão entre objetos, físicas e até mesmo inteligência artificial. Lewis termina por mencionar que em contraste a este tipo de componentes, o conteúdo do jogo, modelos específicos e texturas, a forma com que os objetos interagem uns com os outros, são componentes do jogo em si e já não fazem parte da Game Engine[8].

Unity Engine

Unity é um Game Engine, Cross-Platform, lançada em 2005 com o âmbito de democratizar o desenvolvimento de videojogos, tornando estes mais acessíveis aos utilizadores (Fig 2.1). A funcionalidade Cross-Platform, permite criar aplicações para diversas plataformas, incluindo telemóvel, computador, consolas e realidade virtual[9].

Unity permite aos utilizadores criar videojogos ou experiências em 2D ou 3D e a Game Engine oferece a opção de toda a programação do videojogo ser feita em c# (ao contrário da linguagem de programação popular no mundo dos videojogos, c++). Para jogos 2D o Unity oferece a opção de importar todo o tipo de Sprites e um editor próprio para a conceção de um nível 2D, enquanto para jogos 3D é suportado compressão de texturas, reflexão de ambiente, mapas de sombras e todo o tipo de efeitos pós processamento[10].

Marie Dealessandri descreve a ferramenta unity como o pilar para o desenvolvimento de jogos para projetos Indie e jogos de telemóvel. Dealessandri acrescenta que em Setembro de 2019, 52% de todos os 1000 videojogos de telemóvel mais populares, utilizavam a técnologia unity[11].

Uma das grandes vantagens desta ferramenta é o seu preço. Unity é de uso totalmente gratuito, na condição que os jogos desenvolvidos são gratuitos ou não ultrapassam os

100 mil euros anuais nos últimos 12 meses. O mesmo oferece também uma versão pro, geralmente apropriada para utilizadores que decidam realizar um projeto comercial de larga escala[12].

"Não estaria onde estou hoje, se não fosse o Unity."[13]

William Chyr's, criador e artista do videojogo de sucesso Manifold Garden, descreve Unity como a ferramenta que o possibilitou entrar no mundo dos videojogos, graças à sua usabilidade e simplicidade. O mesmo termina por dizer que a ferramenta é excelente também quando usada como ferramenta de prototipagem[13].

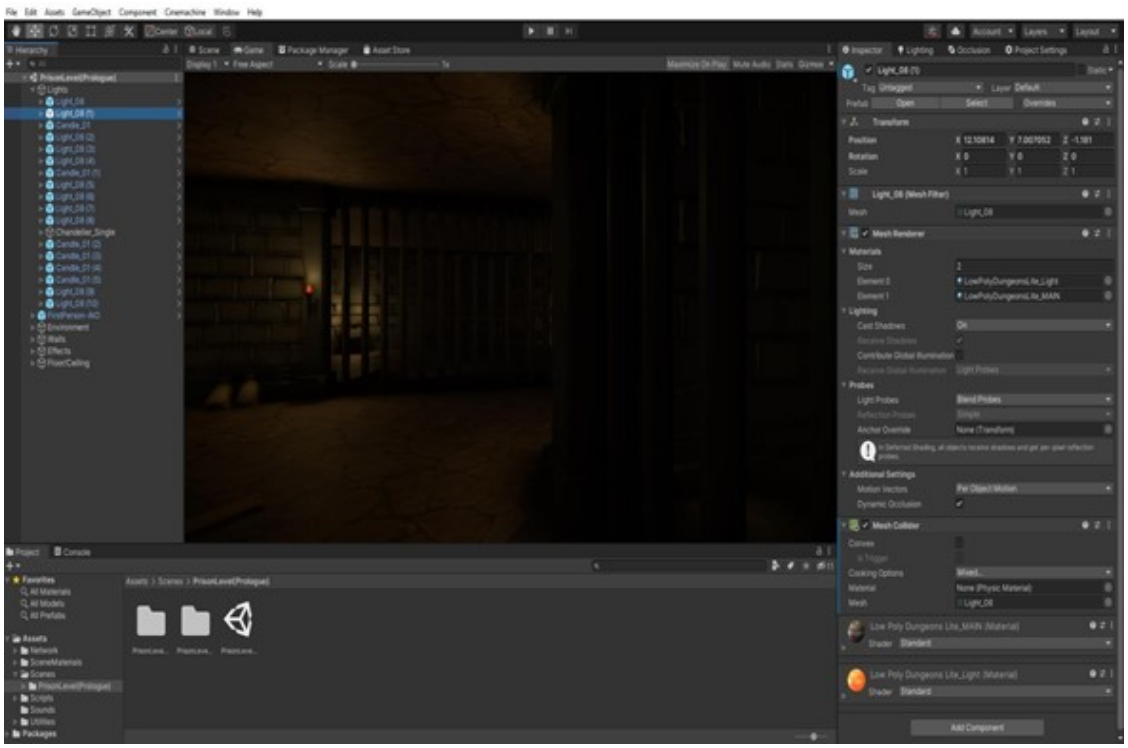


Figura 2.1: Unity Engine Editor

Unreal Engine

Unreal é um Game Engine, desenvolvido em 1998, com o intuito de suportar o jogo Unreal. Esta versão de Game Engine combinava vários componentes, incluindo sistemas de renderização, sistemas de colisão, inteligência artificial, networking, Scripting e gestão de ficheiros (Fig 2.2). Este oferece também um sistema com um alto nível de portabilidade, suportando também várias plataformas tais como computador, telemóvel, consola e realidade virtual[14].

Unreal oferece suporte nativo para código em C++ ou código em blocos, sendo que Sanders considera que esta funcionalidade de código em blocos combinada com o alto nível gráfico que esta ferramenta proporciona, impulsionou a atração e o uso desta, sendo que alguns dos jogos mais jogados no computador atualmente funcionam com base em Unreal engine[14].

Em 2014, a empresa responsável pelo Unreal, Epic Games, disponibilizou licenças gratuitas para escolas e universidades, de modo a auxiliar alunos que frequentassem áreas que beneficiassem deste engine, sendo que mais tarde em 2015 esta licença tornou-se irrelevante, pois o Unreal acabou por ser tornado público e de acesso gratuito, tendo contudo, uma comissão de 5% sobre os todos ganhos de produtos comerciais[15].

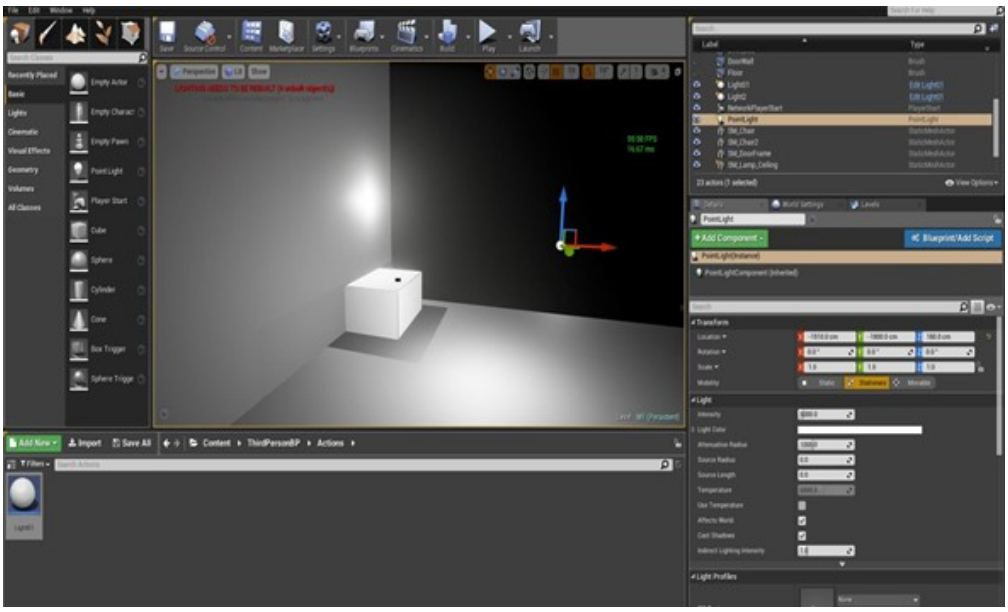


Figura 2.2: Unreal Engine Editor

2.1.3 Inteligência Artificial

Desde cedo o ser humano tentou automatizar tarefas e criar seres "inteligentes". John MacCarthy, define inteligência artificial como *"A ciência e engenharia de criar máquinas inteligentes"*[16]. Desde a criação de robôs, a carros que conduzem sozinhos, inteligência artificial revolucionou o mercado e os consumidores[17].

A Artificial Intelligence (AI) no mundo dos videojogos e no mundo académico, possui diferentes objetivos. No mundo académico, a AI é utilizada para resolver problemas reais e provar teorias sem necessitar de muitos recursos, enquanto que no mundo dos videojogos, a AI tem como objetivo popular os NPCs de modo a fazer com que estes pareçam inteligentes para o jogador e assim providenciar um ambiente divertido ou educacional para o mesmo[18].

A inteligência artificial é dividida em 3 ramos:

- ***Artificial Narrow Intelligence*** - Este tipo de AI é referido quando o sistema está treinado para refazer as mesmas funções únicas. Este tipo de AI não possui aprendizagem própria e só realiza aquilo para que foi preparado. Tipos desta AI incluem carros autónomos, mapeamento de doenças automático e videojogos[19].

- ***Artificial General Intelligence*** - Com este nível de AI, já se pode esperar um comportamento mais avançado, sendo que este nível é competente o suficiente para criar máquinas para trabalhar como humanos. Desde reconhecimento, a imaginação e analogias, este tipo de AI já possui um acesso a dados que lhe permitem ter uma autoaprendizagem, até um certo ponto[19].

- ***Artificial Super Intelligence*** - O último nível de AI é ainda uma tecnologia do futuro, concebida teoricamente para ultrapassar a inteligência humana. Esta a ser estudada para ser excelente a tomar decisões, em arte e até mesmo em relacionamentos, contudo ainda é muito cedo para prever o desenrolar deste nível[20].

2.2 A importância do level design num videojogo

Muitos autores consideram que um bom Level Design é vital para estabelecer a historia do jogo e manter os jogadores imersos. Um mau Level Design, pode por sua vez, arruinar completamente a imersão de um videojogo, independentemente do quão bom seja o grafismo ou o detalhe dos modelos. A indústria dos videojogos evoluiu bastante nos últimos anos, mas algo que sempre prevaleceu é o fato de que se as escolhas efetuadas durante o Level Design forem fracas, o interesse pelo o videojogo tende a ser menor. Ao observar-se a (Fig 2.3), repara-se em alguns pontos fulcrais que se devem ter em conta, durante a conceção de um videojogo.

Independentemente se o videojogo é de um só jogador ou multijogador, o Level Design precisa de ser adequado e bastante vincado no estilo do mesmo. Se olhar-mos para a famosa série do "Call of Duty"[21], o diretor David Vonderhaar, descreve que para a concepção dos níveis de multijogador do videojogo, o mesmo gosta de dividir o mapa em diferentes zonas específicas de modo a gerar diferentes tipos de conflitos entre os jogadores. Não só isto proporciona um estilo de Gameplay diferente em cada etapa do mapa, mas também um sentimento de frescura cada vez que um jogador trocava de área. Vonderhaar descreve que normalmente em mapas de tamanho médio, o mesmo escolhia dois lugares propícios a que os jogadores se escondessem lá, à espera de apanhar outros de surpresa. Descreve também que normalmente todos os mapas tinham lugares *"hotspot"*, lugares estes que o mesmo editava de modo a que fossem mais amplos e que permitissem um estilo de combate mais aberto e com mais movimento[22].

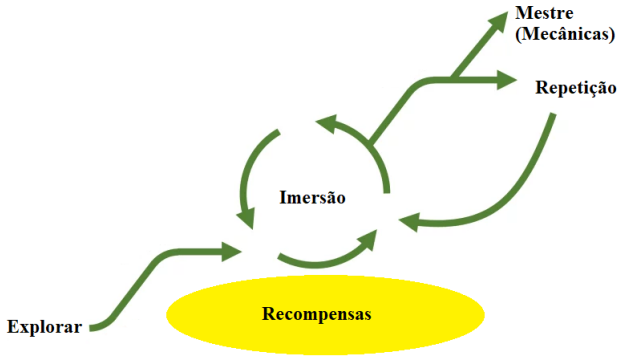


Figura 2.3: Pontos fulcrais na conceção de um videojogo(Foco no jogador)(Adaptado de [23])

2.2.1 Planeamento de níveis

O autor John Feil defende que planear os níveis de um videojogo é um dos passos mais importantes para a criação do mesmo. Desde o planeamento em papel aos teste em terreno, o autor defende que é importante reter o que o jogador pode ou não pode fazer e o seu objetivo no mapa, sendo estes dados importantíssimos para o desenvolvimento dos níveis[7].

Um videojogo com um bom nível tem de ser divertido de navegar. Ao analisar-mos o videojogo "Mirror's Edge"[24], em que ingressamos na pele de uma jovem que faz *parkour*, todo o nível está moldado em torno do *parkour*, tornando-se intuitivo para o jogador o que fazer em cada obstáculo. Mas existe uma diferença entre intuitivo e divertido. Enquanto o progresso pelo nível é efetuado através deste modo de *parkour*, o videojogo possui áreas escondidas do jogador e objetivos diferenciados com diferentes alternativas, de modo a adicionar profundidade ao Gameplay e por sua vez, fazer o jogador sentir-se como que perdido e criar este sentido de preocupação.

O autor John Feil acredita que um bom Level Design não necessita de depender de palavras para contar a história[7]. Com isto, podemos utilizar como exemplo o grande sucesso do videojogo "Bioshock"[25]. Em muitos do mapas (Fig.2.4), o cenário estava populado de assets narrativos, tais como posters, *graffitis*, danos nas estruturas, entre outros, fazendo com que o jogador, quase que intuitivamente, já conseguisse imaginar vários cenários do porque que estaria ali, mas também do que estaria para vir.

Figura 2.4: Videojogo Bioshock

2.2.2 Desenrolar de níveis

Um bom Level Design consegue ter a capacidade de guiar o jogador, mas sem que pareça óbvio, ou seja, um mapa tão bem elaborado que o jogador acredita que está a descobrir algo por ele, mas na verdade, as decisões artísticas levaram-no aquela decisão. Um dos exemplos mais utilizados no videojogos atuais é a criação de pistas que despertam a curiosidade do jogador. Podem estas ser luzes mais fortes, uma parede partida, uma rocha mal colocada propositadamente que leva à entrada de uma gruta, entre outras. Pistas como estas são o suficiente para despertar o interesse do jogador, levando este a explorar essa zona em busca de algo diferente, sem que o jogo lhe indique que existe algo ali.

Mark Cerny, um veterano da área de Level Design, acredita que o jogador deve ser abordado com um número variado de objetivos, que podem ser concluídos em qualquer ordem. Cerny acrescenta que este tipo de abordagem oferece, mesmo que não seja real, um sentido de controlo ao jogador[26]. O mesmo utilizou estas técnicas na maioria dos seus trabalhos, incluindo no grande sucesso da Sony[27] "Ratchet & Clank"[28].

2.2.3 Recompensar o jogador

Em qualquer videojogo, caso exista uma área que pareça importante ou contém algum tipo de desafio para lá chegar, o jogador deve ser recompensado, sendo estas recompensas variadas, tais como munição extra, troféus, armas ou poderes novos, entre outros (Fig.2.5). Isto torna-se ainda mais necessário, quando o jogador não depende dessas mesmas áreas para concluir o nível, oferecendo assim ao jogador uma recompensa pela exploração.

Alex Mandryka, diretor de Level Design no videojogo "Assassin's Creed"[29], defende que este fator de recompensa ou surpresa, no caso do desafio, é a verdadeira essência de diversão. O mesmo afirma que em termos de Level Design, o fator surpresa gera um momento que ensina ao jogador algo novo, seja isto a utilização de uma mecânica num novo formato que em combinação com o novo fator, modifica a jogabilidade[30].

Com isto, um bom Level Design está sempre a ensinar ao jogador algo novo, seja esta uma habilidade, uma variação de Gameplay ou a utilização de um item de uma forma diferente. Raph Koster explica que a mente humana aprecia processar informação

sobre o mundo em padrões, para mais tarde ser mais fácil de processar. Em termos de jogabilidade isto pode ser traduzido para que grande parte da diversão é gerada em torno da aprendizagem de várias mecânicas. Koster afirma também que se as mecânicas e padrões de jogabilidade forem muito simples, os jogadores podem ficar rapidamente aborrecidos e parar de jogar[31].

Figura 2.5: Videojogo The Witcher 3

2.2.4 Testes aos níveis

John Feil, defende que um dos passos mais importantes, é sem dúvida ter uma fase de testes sólida [7]. Desde disponibilizar mapas para serem testados por terceiros, a avaliações das estruturas, a fase de testes irá permitir analisar erros de conceção e alguns pontos onde as mecânicas não são tão adequadas, seja isto na forma de demasiados inimigos num certo local, ou até mesmo verificar se o nível não está a desenrolar da forma esperada (exemplo Fig. 2.6).

O autor John Feil, refere-se a esta fase como a de polimento do videojogo. O mesmo defende que durante está importantíssima fase, diversos tipos de teste externos devem ocorrer, de modo a garantir que a maioria dos Bugs que possam prejudicar a experiência do jogador, sejam corrigidos. Feil também defende, que após outros indivíduos testarem os níveis, todo o seu Feedback deve ser tomado em especial atenção com as necessidades

do nível, para garantir que as premissas do mesmo não são alteradas[7].

Figura 2.6: Protótipo The Abyss

2.3 Level design num videojogo multijogador

Do ponto de vista de um videojogo de um só jogador, as regras convencionais divergem do que é requerido para um videojogo multijogador. Esta nova camada, adiciona obstáculos que antes não necessitavam de ser abordados, tais como ajustes de dificuldade dinâmicos, jogabilidade, mecânicas e interações sociais.

Pascal Luban, Level Designer de multijogador nos videojogos "Splinter Cell"[32], afirma que durante a criação do Level Design de um videojogo multijogador, é necessário ter várias considerações em conta. Estas variam desde as capacidades técnicas, ou seja, quantos jogadores em simultâneo, quantos efeitos especiais por zona, eventos dinâmicos, entre outros, ao tamanho dos mapas de modo a garantir uma fidelidade enorme na jogabilidade entre os jogadores. Luban afirma também que estas considerações não acontecem em videojogos de um só jogador, pois tudo é desenhado à medida, mas em multijogador, tudo é possível[33].

Durante o planeamento do Level Design de um videojogo multijogador, é fundamental definir bem o tipo de multijogador que este suportará, pois o videojogo será desenhado em volta do mesmo.

2.3.1 Co-op

Quando um videojogo é categorizado como Cooperative Video Game (Co-op), o mesmo está suportado para o modo cooperativo. Com este modo, o videojogo permite ao jogador convidar um ou mais amigos para a sua sessão, na qual podem todos partilhar a experiência de explorar o mundo, combater inimigos, completar missões, entre outros. Este modo de jogo é normalmente focado em videojogos de categoria Player Versus Environment (PVE), onde o âmbito do jogadores é de cooperarem entre si e não uns contra os outros[34]. Um exemplo é o famoso videojogo "Assassin's Creed Unity"[35] (Fig.2.7), na qual o jogador pode conetar-se com até quatro jogadores e jogarem em simultâneo.

O modo Co-op, surgiu primeiro de forma local, ou seja, na mesma máquina física. Contudo, a tecnologia evoluiu e o Co-op progrediu com ela, sendo atualmente possível jogar videojogos em Co-op de rede, sem que seja necessário os jogadores estarem perto um do outro, ou na mesma máquina física[34].

Do ponto de vista de Level Design, os videojogos deste tipo requerem uma atenção diferente durante o desenvolvimento, pois, como podem suportar um ou mais jogadores, é importante que os cenários estejam em conformidade para tal. Com isto, todo o nível é pensado para funcionar com a vertente do possível Co-op, mesmo que o jogador decida ou não utilizar essa função.

Figura 2.7: Videojogo Assassin's Creed Unity

2.3.2 Multijogador em rede

Multijogador em rede é atribuido a videojogos, que normalmente estão pensados para serem jogados inteiramente em rede. Este tipo de videojogos encoraja, ou em alguns casos tem como única opção, que o jogador jogue sempre *online*[36]. Apesar do estilo Co-op ser considerado uma categoria de multijogador, o mesmo é distinto dos outros estilos.

Este modo de multijogador, ao contrário de Co-op, possui uma vasta gama de categorias associadas, sendo que estas categorias mudam completamente a jogabilidade e o estilo do videojogo. Categorias estas que podem variar desde de First-Person Shooter (FPS), Massively Multiplayer Online Game (MMO) ou até mesmo Multiplayer Online Battle Arena (MOBA). No caso do Co-op, normalmente o jogador pode, ou não, aliar-se a outros jogadores e em conjunto jogarem contra o computador, ou seja, estilo PVE. Mas, com a introdução do multijogador em rede, uma nova categoria surge, sendo esta chamada de Player Versus Player (PVP), em que o jogador pode agora enfrentar outros jogadores, ao invés de estar limitado a enfrentar só o computador[36].

Do ponto de vista de Level Design, o artista deve ter muito em consideração que tipo de estilo multijogador, o videojogo irá tomar. Ao contrário do Co-op, em que o mapa pode ou não ser jogado por mais que um jogador em simultâneo, neste estilo já é esperado que vários jogadores o façam (exemplo fig.2.8).

O autor Benjamin Bauer's, afirma que muitos artistas tendem a modificar as mecânicas do videojogo para fazer sentido nos níveis, quando na ótica do mesmo, deveria de ser o contrário[37]. Assim, definir previamente se o videojogo incluirá PVP, acaba por salvaguardar, do ponto de vista artístico, as mecânicas do videojogo. Com isto, o Level Designer, supondo um videojogo do estilo FPS, terá de ter em consideração outros aspetos de jogabilidade, tais como ângulos de tiro, espaços de luta geral, pontos mortos pelo mapa que permitam emboscadas noutros jogadores, entre outros, de modo a garantir um ambiente divertido e equilibrado para os jogadores.

Figura 2.8: Videojogo World of Warcraft

2.4 Arte dos níveis

A jogabilidade de um nível, desempenha um papel crucial para o impacto sobre o jogador, mas a arte do mesmo partilha desta grande responsabilidade. Durante a conceção de um nível, o aspeto visual é tão fulcral, que este possui características que podem auxiliar o jogador a tomar decisões. Desde luzes que simulam ambiente, a texturas que podem orientar o jogador, a arte do nível é uma etapa que não pode ser ignorada.

2.4.1 Luzes

Luz, num videojogo, ocupa um cargo enorme, sendo este o cargo de conseguir dar informações ao jogador, sem que este necessite de texto. Este tipo de informação pode surgir desde uma área mais escura, em que o jogador apercebe-se que pode optar por ter uma jogabilidade mais lenta e cautelosa, a áreas brilhantes e bonitas em que o mesmo fica completamente emergido.

A revista "Pluralsight"afirma que a luz num videojogo, é o fator mais determinante em termos visuais. A mesmo afirma que a luz pode ser a razão para um videojogo ser um sucesso visual[38].

Ao analisar-se "Red Dead Redemption 2"[39], um dos grandes sucessos da última década, verificou-se que o trabalho realizado em torno do ambiente e das luzes é simplesmente único (Fig.2.9). Luke Reilly, descreve o trabalho artístico de "Red Dead Redemption 2"como fantástico, sendo esta uma das razões para a pontuação tão elevada que o mesmo adquiriu na maioria das *reviews*[40].

A maioria dos Game Engines divide luzes em quatro variações distintas, sendo estas:

- ***Point Light*** - Termo dado a uma luz que consegue emitir luz, por igual, em todas as direções. De acordo com o Unreal, este tipo de luz é recomendado para simular lâmpadas convencionais[41].

- ***Spot Light*** - Termo dado a uma luz que consegue emitir luz com o formato de cone. De acordo com o Unity, este tipo de luz é recomendado para simular lanternas, faróis de veículos, entre outros.[42].

- ***Directional Light*** - Termo dado a uma luz que consegue emitir luz para a cena inteira. De acordo com o Unreal, este tipo de luz é recomendado para simular o sol ou a lua[41].

- ***Area Light*** - Termo dado a uma luz que consegue emitir luz num formato rectangular, mas só para uma das faces. De acordo com o Unity, este tipo de luz é recomendado para simular luzes de rua ou até mesmo pontos mais pequenos como o interior de uma casa[42].

Figura 2.9: Videojogo Red Dead Redemption 2

2.4.2 Texturas

O Unity descreve texturas como uma imagem aplicada sobre uma superfície[43]. Texturas, tal como a luz, auxiliam o jogador do ponto de vista visual. Contudo, as texturas focam em providenciar outro tipo de informação, tal como, orientação e navegação.

O autor Benjamin Bauer descreve que as texturas, auxiliam o jogador a perceber onde está situado. O mesmo acrescenta que em videojogos táticos, pontos de partida das diferentes equipas possuem diferentes cores, tipos de materiais e outros detalhes, de modo a orientar os jogadores pelo mapa[37].

Texturas podem ser divididas em diferentes tipos, dependendo das necessidades e o porque de terem sido criadas. O Unity, tal como o Unreal, possui suporte para diferente tipos de texturas[44]:

- ***Default texture*** - Tipo mais utilizado de textura, de acordo com o Unity. Este tipo de textura é normalmente utilizado com outros parâmetros de modo a colocar texturas rápidas em cena[44].
- ***Normal map*** - Normal map é um tipo de textura utilizada para introduzir mais detalhe na textura original. É normalmente utilizado sobre uma textura já existente[44].
- ***Lightmap*** - Lightmap é uma textura pré renderizada, que contém informação sobre espectros de luz. Com este tipo de textura, podemos simular sombras, ricochete de luzes, mas de forma a que estejam estampados na textura, sendo que não necessita de processamentos futuros[44].

2.5 Otimizar um videojogo

Criar um nível bonito e divertido dentro de um videojogo, é algo único, mas para o fazer corretamente são necessárias algumas técnicas de otimização. Atualmente, no mercado de videojogos, temos o exemplo do famoso videojogo "Cyberpunk2077"[45], que apesar da expetativa de toda a gente, teve um lançamento bastante atribulado[46]. Este deve-se ao fato do videojogo ter sido bastante apressado durante o lançamento, não tendo sido permitido aos Level Designers otimizarem grandes partes dos níveis, originando críticas

pelo lado dos jogadores, que demonstraram o seu descontentamento na página de críticas do videojogo[47].

O autor Ben Garney afirma que otimizar um videojogo é um jogo de trocas de recursos[48]. O mesmo descreve que a otimização passa por 5 etapas:

- ***Benchmark*** - O autor refere-se a esta fase como a fase de testes aos níveis existentes. O mesmo adiciona que esta fase facilita a correção dos níveis, sendo que o Feedback é imediato[48].
- ***Detection*** - O autor refere-se a esta fase como a fase de deteção, para verificar o que precisa de ser otimizado[48].
- ***Solve*** - O autor refere-se a esta fase como a fase de resolução, sendo aqui implementadas as correções para os erros detetados na etapa anterior[48].
- ***Check*** - O autor refere-se a esta fase como a fase de verificação, sendo aqui analisadas as imlementações da etapa anterior e verificar se as mesmas tem o efeito pretendido[48].
- ***Repeat*** - Esta é a ultima fase que volta a dar início a este ciclo, sendo que o autor adiciona que corrigir um erro muitas vezes pode desancadear outro, e daí, ser importante rever tudo e voltar a corrigir se necessário[48].

2.5.1 Importância da otimização num videojogo

Com o avanço tecnológico, os requisitos para os videojogos alteraram. Desde os tempos em que 30 imagens por segundo eram considerados a base, aos tempos atuais em que o jogadores esperam no mínimo velocidades de 60, ou mais, imagens por segundo[49], a otimização nos videojogos nunca foi tão importante.

Não só a otimização do videojogo é responsável pela quantidade de imagens por segundo que o utilizador consegue jogar (assumindo que este possui os requisitos recomendados), mas também a responsável pela boa utilização dos componentes de cada computador (GPU, CPU, RAM e disco rígido).

O autor Ben Garney afirma que otimizar é um trabalho que pode não terminar, sendo a eficiência uma necessidade[48].

Com o recente lançamento de "God of War"[50], na plataforma PC, podemos verificar o impacto positivo que uma boa otimização oferece ao videojogo. Desde críticas extremamente positivas[51], responsividade excelente e uma tremenda gama de compatibilidade entre diferentes máquinas, "God of War"foi um sucesso de vendas que deve ser utilizado como exemplo para quem trabalha com otimizações[52].

2.5.2 Técnicas de otimização

No processo de otimizar um nível de um videojogo, algumas técnicas são essências, tais como:

Occlusion Culling

Occlusion culling é referido como o processo que permite ao utilizador, indicar à camera do jogador que tudo o que não estiver a ser renderizado de momento, é descartado (exemplo Fig. 2.10). Com isto, todo o terreno, objetos, entre outros, que se situarem atrás da camera do jogador, não estarão a consumir recursos, pois por sua vez, são descartados[53].

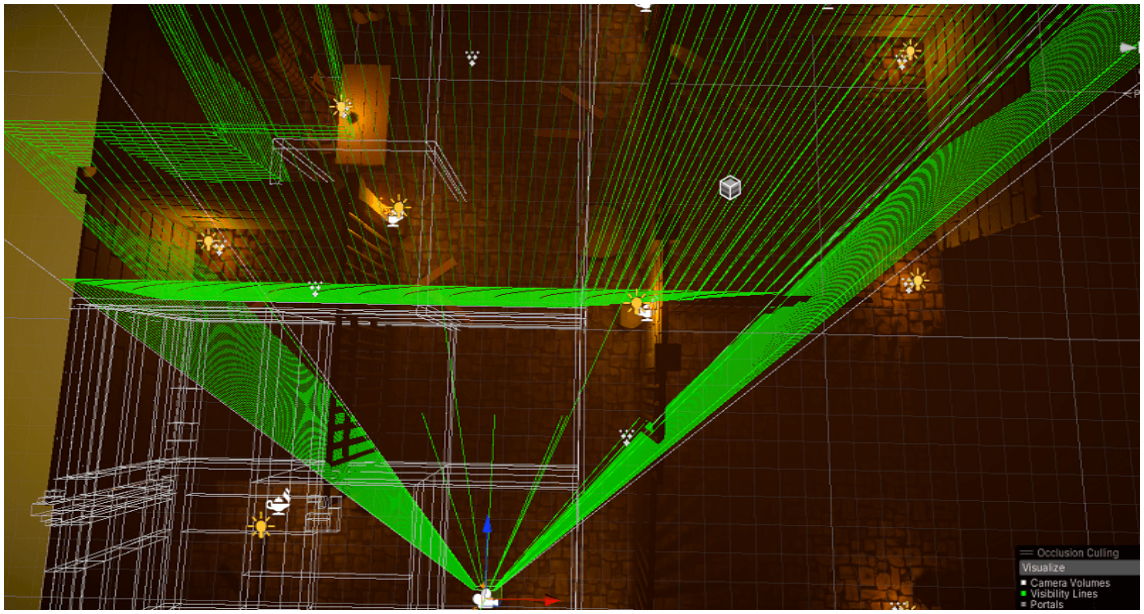


Figura 2.10: Occlusion Culling

Bake lighting

Sempre que possível, deve ser utilizado luzes "*baked*". Ao utilizar este modo de luz, quando a cena é processada, a luz é transformada em textura e guardada nos "*lightmaps*"e a sombra dos objetos estáticos é transformada numa textura também (Fig. 2.11). Com isto, durante o decorrer do videojogo, estas luzes e sombras já não são mais processadas, pois agora são texturas, poupando assim bastantes recursos[54].

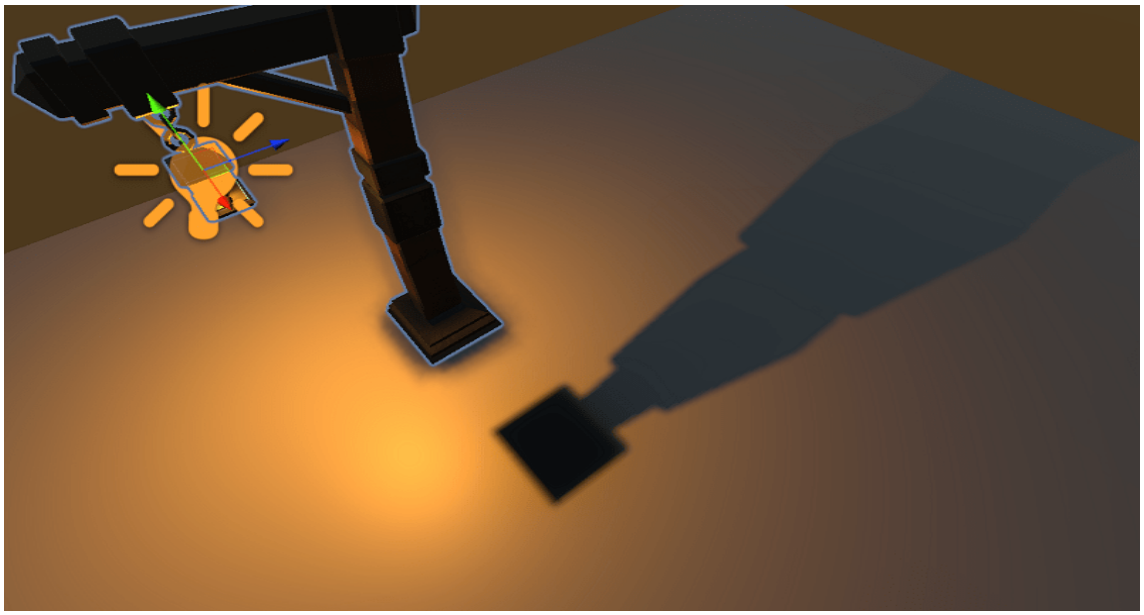


Figura 2.11: Lightmap

Inspector

Durante o processo de otimização, a utilização da janela *Inspector* é crucial, pois esta ilustra dados da cena (Fig. 2.12), em todas os segmentos de imagem, com a sua devida informação e consumo. Esta janela permite ao utilizador verificar que aspeto do videojogo está mais fragilizado, em termos de desempenho[55].

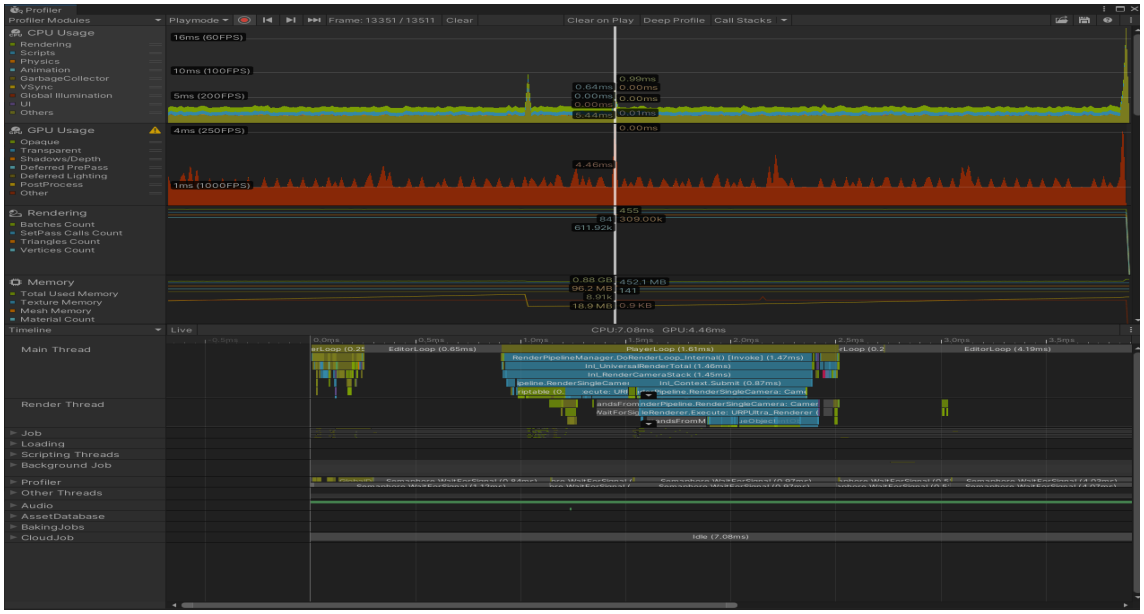


Figura 2.12: Profiler

Combine meshes

Por ultimo, uma boa prática é combinar as *meshes*(camada dos objetos constituída por vértices e arrays triangulares (Fig. 2.13)) de vários objetos colocados perto um do outro em cena, de modo a poupar recursos. Cada vez que existe uma *mesh* nova em cena, é mais um processo a ser executado e com isto, reduzindo o número de *meshes* totais, o número de recursos necessários para correr a cena, é reduzido também[56].

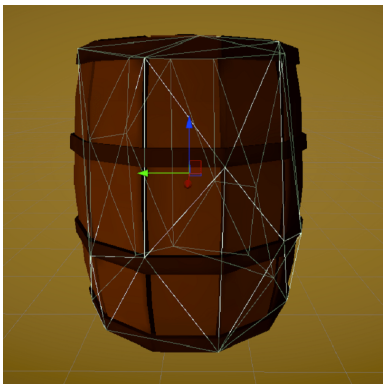


Figura 2.13: Mesh

Capítulo 3

Planeamento

Para este projeto foi utilizado o método de *Practice-Based Research*. Este método foca-se no processo de aprendizagem, usando como base outras áreas já investigadas. Apresenta também características muito positivas na aquisição de novos conhecimentos e é atualmente um dos métodos mais utilizados, devido às suas altas taxas de sucesso, quando posto em prática[57].

O projeto "The Abyss" é um videojogo que incorpora elementos RPG em conjunto com elementos TPS. Os dois conceitos já estão bastante estudados e conhecidos, mas não quando utilizados em conjunto. Com isto, utilizando o método acima referido, conseguiu-se conciliar o conhecimento existente de duas áreas e acabar com um projeto rico em várias vertentes.

3.1 Projeto a Desenvolver

"The Abyss" consistire num videojogo com elementos RPG e TPS, compatibilidade multijogador e uma narrativa e atmosfera imersiva. Com isto em mente, decidiu-se que o uso do Unity como motor de desenvolvimento iria trazer diversas vantagens, uma das quais a linguagem de programação. O Unity funciona com base de *scripts* em C# e permite integração de *API's* (Network for gameobjects), que permite construir manualmente os pilares para o suporte multijogador. Uma das grandes diferenças que o Unity proporciona, sobre os outros motores de desenvolvimento, é a sua modularidade e grande nível de opções

ao desenvolver o jogo.

O Unity permite que o utilizador consiga editar todos os recursos em cena, verificando usos de memória, utilização da GPU e CPU, tipos de processos que a GPU necessita de tratar para executar uma função. Isto oferece um trunfo para o utilizador que pode otimizar a cena de tal modo que o jogo não só fica mais apelativo visualmente, mas como funciona melhor.

A linguagem de programação teve um enorme peso também na escolha do motor de desenvolvimento, sendo que foi optado por utilizar C#, que, devido às suas características de Programação orientada a objetos, adequava-se às necessidades do projeto. Mecânicas de jogo, jogabilidade e inteligência artificial utilizam C# para o seu desenvolvimento.

Para conciliar o projeto entre o grupo, foi utilizado recurso a uma ferramenta *Github Desktop*, que permite acesso a um repositório partilhado pelo grupo. *Github Desktop* funciona com base Git e exige que após alterações cruciais ao projeto, seja efetuado um *push* (enviar informação para o repositório), de modo a sincronizar todos os membros do grupo.

3.2 Implementação do Projeto

Até ao momento, o projeto já possui alguns mecanismos de movimentos, animação, Level Design e Networking implementados. Muitos destes mecanismos estão implementados como base para o futuro e utilizados de momento para testes.

O projeto já possui um nível introdutório, utilizado como o tutorial para o jogador, ao iniciar o videojogo pela primeira vez. Esta cena foi concebida através uma estrutura lógica, seguindo um planeamento para o desenrolar da narrativa, de modo a introduzir o jogador à mesma, sem que este fique perdido e desorientado. A cena sofreu também bastantes remodelações e otimizações, para garantir extrema fiabilidade em qualquer tipo de sistema e garantir que todos os recursos do sistema estão a ser utilizados o melhor possível.

3.3 Testes

Durante algumas etapas do projeto, foram disponibilizadas versões Alpha e Beta, a um número selecionado de indivíduos, de modo a obter Feedback para o projeto. Com este Feedback, pode-se alterar vertentes do projeto, missões ou mesmo narrativa para que o videojogo alcance níveis de satisfação esperados. Todavia, permitir que outros indivíduos testem o projeto providencia informação sobre otimizações necessárias e alguns Bugs que possam ter escapado.

3.4 Cronograma

O seguinte cronograma representa, não só tarefas individuais, mas também as tarefas do projeto como um todo. As tarefas apresentadas com a cor laranja, representam tudo o que foi feito para o projeto como um todo, sendo que as amarelas representam as tarefas individuais.

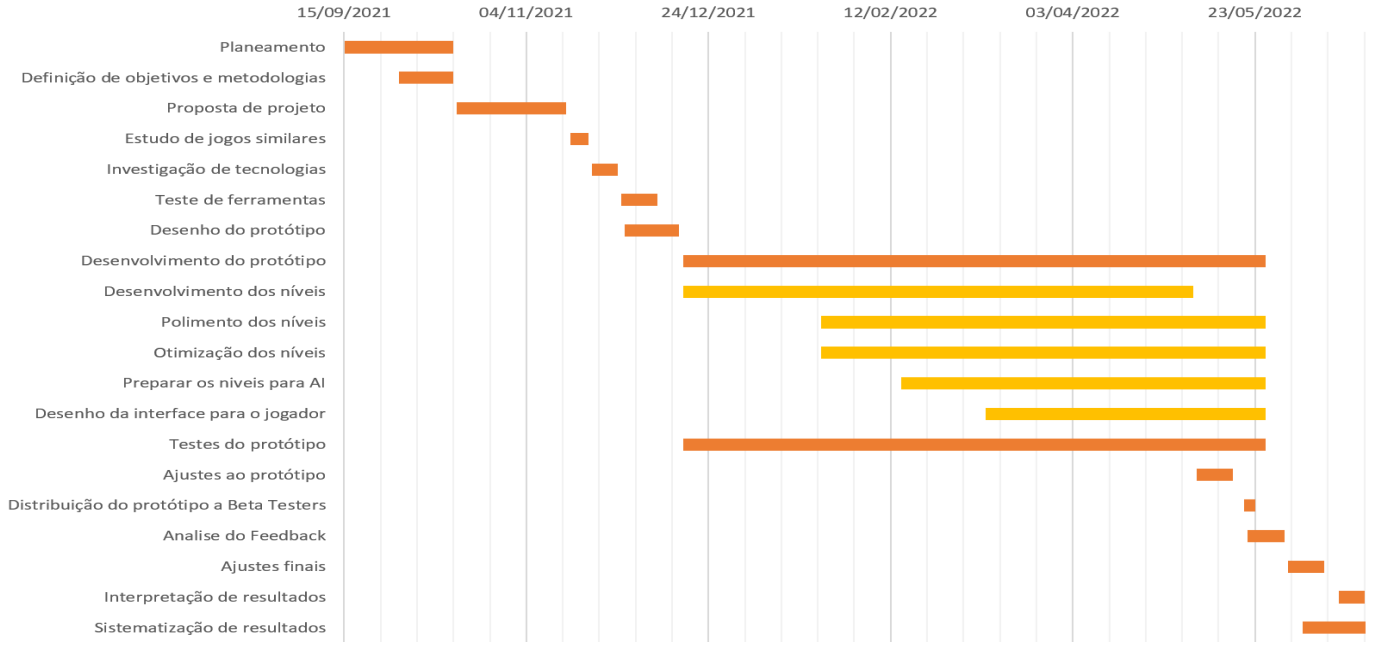


Figura 3.1: Cronograma

Como referido no cronograma, o desenvolvimento começa por a criação do níveis, onde por sua vez colide com a otimização e polimento dos mesmo. Em fase de desenvolvimento mais avançadas, é dado inicio ao desenvolvimento das variáveis dos níveis para serem utilizados por a inteligência artificial. Por ultimo, é desenvolvido a interface para o jogador.

3.5 Requisitos

No decorrer deste projeto, alguns requisitos podem variar, consoante limitações técnicas. Mas para o desenvolvimento de níveis num videojogo multijogador, é esperado:

- **R1** - Definir e implementar os níveis do videojogo.
- **R2** - Definir o estilo visual do videojogo.
- **R3** - Desenvolver caminhos lógicos para orientar o jogador pelos níveis.
- **R4** - Definir localização de mecânicas, sons, efeitos especiais e formas de jogabilidade.
- **R5** - Desenvolver áreas de combate, com suporte dinâmico de até quatro jogadores.
- **R6** - Definir tipos de interação entre o jogador, outros jogadores e o mundo do videojogo.
- **R7** - Desenvolver caminhos lógicos e tipos de jogabilidade para a inteligência artificial.
- **R8** - Criar sistemas de otimização durante todos os níveis, de modo a garantir uma *performance* estável.
- **R9** - Desenvolver uma interface gráfica básica para o jogador.

3.6 Recursos

Game Engine

- **Unity 3D** - Motor de desenvolvimento de jogos multiplataforma, desenvolvido por Unity Technologies.

Software

- **GitHub Desktop** - Aplicação baseada em Git, com GUI, para a representação visual de repositórios.
- **Blender** - Software de modelação e animação 3D.

API

- **Network for gameobjects** - API de networking, desenvolvida para o Unity, de modo a suportar camadas de multijogador.

Capítulo 4

Desenvolvimento

No decorrer deste capitulo, será relatado todo o processo envolvente na criação e desenvolvimento de níveis, no videojogo "The abyss". Será apresentado com foco nas áreas de Level Design, onde será descrito todo o processo de planeamento, o tipo de arte escolhido, criação dos níveis, a interface para o jogador e as otimizações de cena.

4.1 Planeamento dos níveis

O planeamento consiste numa fase fundamental do projeto, onde as ideias são colocadas em prática e os primeiros esboços dos níveis começam. Durante este processo foram discutidas ideias de como seria o estilo prático do videojogo, o seu ciclo de jogabilidade e as mecânicas envolventes. Foi também discutido o estilo ambiente e artístico para os níveis, tal como o motor de desenvolvimento a ser utilizado.

4.1.1 Motor de desenvolvimento

Durante a fase de seleção do motor de desenvolvimento, foram considerados os pontos positivos e os contratempos de cada um destes, de modo a encontrar o mais adequado para o tipo de projeto a ser realizado. Perante tais argumentos, decidiu-se utilizar o Unity, como o motor de desenvolvimento, devido à sua acessibilidade com a linguagem de programação, com o acesso a ferramentas adequadas ao projeto e à sua vasta disponibilidade de Assets.

4.1.2 Estrutura dos níveis

Após várias considerações, decidiu-se a estrutura geral para o videojogo, ou seja, que tipo de níveis e jogabilidade estes iriam conter. Como se pode observar na (Fig. 4.1), o videojogo irá possuir três níveis principais, sendo que, o nível de ação é expandido à medida que o jogador vai progredindo. Todos os níveis estão conectados, mas só os últimos dois permitem ser revisitados, pois são estes que ficam incluídos dentro do ciclo de jogabilidade.

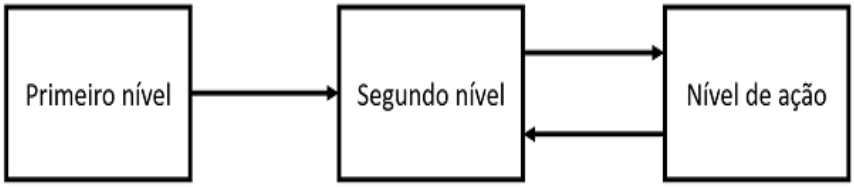


Figura 4.1: Estrutura lógica dos níveis

O projeto "The abyss", consiste num videojogo cooperativo RPG por níveis, logo o ciclo de jogabilidade do videojogo torna-se bastante claro. O jogador irá utilizar o segundo nível como uma espécie de "casa", onde se consegue juntar a outros jogadores, vender itens ou reparar o equipamento e depois prossegue, sozinho ou acompanhado, para um dos níveis de ação, onde consequentemente consegue coletar recursos para trocar ou vender na sua "casa".

Primeiro nível

O primeiro nível funcionará como um nível introdutório à narrativa do jogador. É neste mapa que o jogador é apresentado às mecânicas básicas de movimento, combate e de jogabilidade. Para tal o mapa deve ser linear, lógico e quase a guiar o jogador, para que o mesmo se adapte o mais rápido possível à jogabilidade. Este mapa será também jogável na vertente de um só jogador.

É possível visualizar na (Fig. 4.2), um esboço lógico de jogabilidade do primeiro nível, onde o jogador começa no circulo amarelo e o seu objetivo é ir até ao círculo verde, sendo que a linha azul representa o caminho jogável e o circulo vermelho uma primeira interação para introduzir o jogador às mecânicas de combate. Este esboço servirá como base para a construção do primeiro nível.

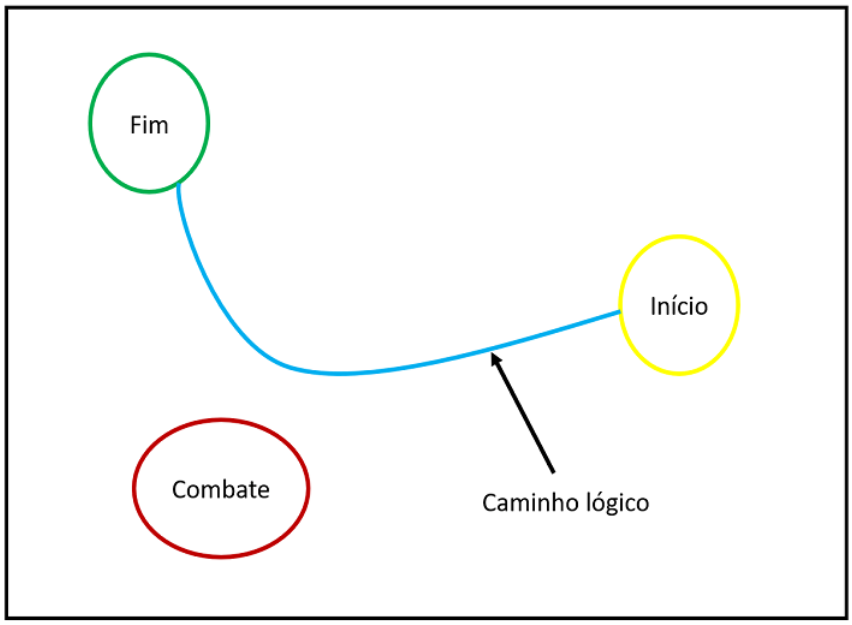


Figura 4.2: Estrutura lógica do primeiro nível

Segundo nível

Após o jogador completar o primeiro nível, o mesmo começa a sua jornada no nível seguinte. Este mapa será o pilar do videojogo, agindo quase como uma "casa"para o jogador, em que o mesmo retorna sempre depois das suas aventuras, seja para se conectar com amigos, vender itens ou materiais, reparar o equipamento ou mesmo para descontrair e apreciar o ambiente.

Este mapa, ao contrário do primeiro, será já um mapa aberto que conterá pontos de interesse e monumentos memoráveis. Os pontos de interesse consistem em despertar o interesse do jogador, fazendo este os visitar. É fulcral criar uma boa atmosfera neste nível, visto que o jogador irá visitá-lo regularmente.

Ao analisar-se a (Fig. 4.3), repara-se que este esboço de nível já contém outras opções para o jogador, tais como pontos de interesse em diversas localizações, com possível integração de objetivos nos mesmos e um caminho lógico, mas não linear. Desta forma consegue-se oferecer um pouco de liberdade ao jogador, mas sempre dentro das condições que o grupo de desenvolvimento do jogo pretende manter.

Por fim, verifica-se também a conexão para o nível seguinte, que servirá também como ponto de retorno mais tarde.

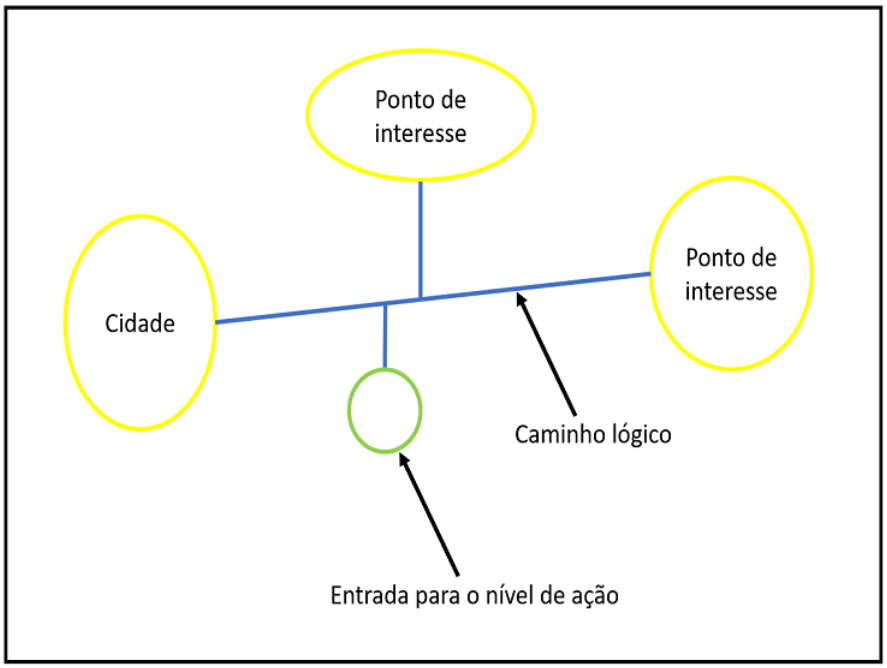


Figura 4.3: Estrutura lógica segundo nível

Nível de ação

Uma vez que o jogador tenha decidido a se aventurar, o mesmo terá então a opção de começar a sua jornada pelos níveis de ação. Os níveis de ação, ao contrário dos anteriores, irão conter e incentivar combate direto com inimigos. Será durante estes níveis que o jogador conseguirá colocar à prova as suas habilidades de combate, ganhar experiência, materiais, itens ou mesmo moedas, para mais tarde fazer vendas e reparações no segundo nível. Estes níveis poderão ser jogados a solo ou em cooperativo, sendo a dificuldade dos inimigos ajustada de acordo com a quantidade de jogadores.

Ao observar-se a (Fig. 4.4), verifica-se a existência de pontos estratégicos de combate e um caminho lógico para o jogador seguir, sendo este orientado através de pontos de interesse que despertem a atenção do jogador. Os pontos de combate tornam-se mais difíceis à medida que o jogador vai evoluindo, dando assim um sentido de progressão e dificuldade. Contudo, na ultima instância de combate, o jogador enfrenta um inimigo mais forte que os restantes e só derrotando este, é que o mesmo consegue passar para o próximo nível, ou retornar ao segundo nível com tudo o que colecionou durante o decorrer da sua aventura.

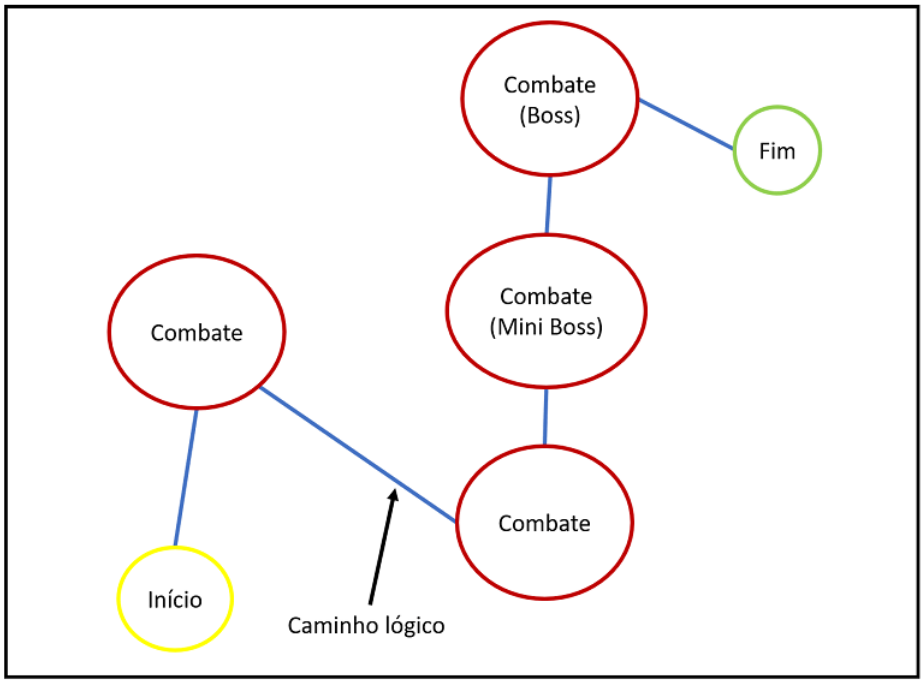


Figura 4.4: Estrutura lógica nível de ação

4.1.3 Arte

Após o planeamento lógico dos níveis, foi necessário definir o estilo artístico do videojogo e o tipo de ambientes em que este se situava, para dar então inicio à fase de desenvolvimento prático. Esta fase consistiu na discussão de temas e recolha artística de localizações, fictícias ou reais, para usar como referência para a construção dos mapas. Foi também definido o tipo de luz, cores e aspeto temático para o videojogo.

Estilo artístico

Após várias discussões, decidiu-se desenvolver o videojogo com o tema LOW-POLY e STYLIZED. Os objetos podem ser modelados em duas vertentes, LOW-POLY ou HIGH-POLY (Fig. 4.5), sendo que como o nome indica, LOW-POLY contêm menos polígonos, fornecendo assim um aspeto mais simples. Já os modelos HIGH-POLY são normalmente utilizados em jogos mais realistas, onde mais polígonos são utilizados para um aspeto mais fidedigno.

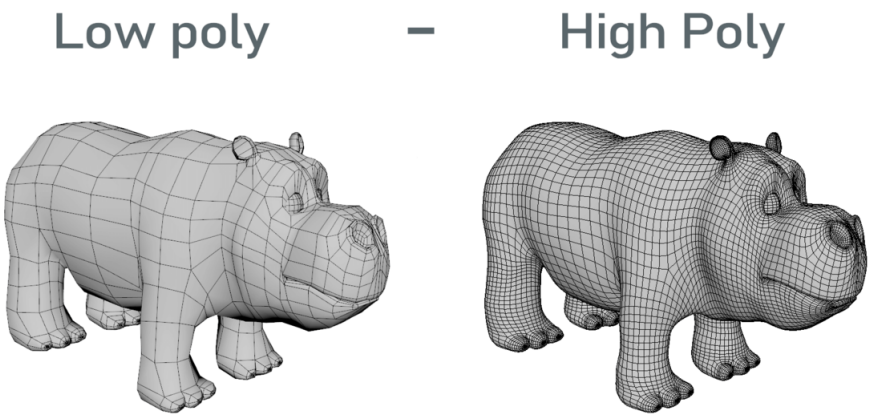


Figura 4.5: Low-poly e high-poly([58])

Decidiu-se combinar o estilo STYLIZED (Fig. 4.6) com LOW-POLY, pois como a arte do videojogo será menos realista e mais orientada ao estilo de fantasia, estes dois estilos são os que fazem mais sentido lógico.

Figura 4.6: Mapa Overwatch 2 - Exemplo STYLIZED([59])

Quanto à temática dos mapas e do videojogo em geral, devido às escolhas referidas anteriormente, às mecânicas já planeadas e à narrativa do jogador, decidiu-se combinar o estilo medieval com uma vertente grande de fantasia, de modo a garantir uma experiência única ao jogador.

Estilo dos mapas

Para a conceção do mundo do videojogo, após o planeamento dos três níveis, necessitou-se de escolher o tipo de arte de cenário para cada um dos níveis, de modo a enquadrar com a narrativa do jogador. Para tal, utilizou-se várias localizações reais e/ou fictícias, num formato de guia.

Para o primeiro nível, utilizou-se como base as prisões da era medieval, que apesar de diferentes dependendo da região em que se localizavam, tinham alguns aspetos em comum, tais como aspeto mais grotesco, construídas no subsolo, celas em paralelo e muitas fontes de luz em todas as divisões.

Enquanto que para o primeiro nível foram utilizadas localizações reais, para os seguintes, devido à narrativa de fantasia do jogador, utilizou-se como base de inspiração e estrutura mapas da série "Made in Abyss"[60], que apesar de contarem uma história diferente, as temáticas enquadram-se bastante bem.

Figura 4.7: Cidade Made in Abyss([61])

A figura (Fig. 4.7) serviu como referência para o desenvolvimento do segundo nível e consequentemente abriu uma possibilidade de arte para os níveis de ação. Uma vez que a narrativa do jogador explica que os níveis de ação funcionam em camadas no subsolo, uma espécie de grutas ampliadas, esta cratera permite fazer uma ligação real entre o mundo do segundo nível e os mapas de ação no subsolo.

Para tal, como referencia, utilizou-se o livro "Viagem ao centro da terra"de Jules Vern [62] e decidiu-se criar biomas diversificados, quase que como uma nova dimensão dentro de cada nível, de modo a fornecer ao jogador uma sensação de novidade e exploração, à medida que o mesmo progride.

Assets utilizados

Uma vez que o estilo artístico e o tipo de arquitetura dos níveis já haviam sido definidos, decidiu-se o tipo de Assets a ser utilizados. Com base no projeto em questão, seria necessário assets com uma vertente LOW-POLY, sendo que para tal, utilizou-se uma combinação de Assets de autoria própria e Assets provenientes da SyntyStudios[63], uma vez que se enquadravam no tema do videojogo.

4.2 Conceção dos níveis

Este capitulo demonstrará todas as etapas de desenvolvimento prático realizadas durante a conceção dos níveis. São abordados aspetos como o de construção de mapas, localização e estrutura de objetivos, estruturação do ambiente para a AI e como se realizou todo o pós-processamento de imagem.

4.2.1 Preparar o motor de desenvolvimento

Uma vez já dentro do Unity, antes de começar qualquer tipo de projeto, definiu-se o tipo de *Pipeline* de renderização, sobre a qual o projeto iria ser trabalhado. Após várias trocas de ideias durante a fase de planeamento, chegou-se à conclusão que a utilização da Universal Render Pipeline (URP) seria a mais adequada, face às necessidades do projeto. A instalação da URP pode ser feita através do *Package Manager*(Fig. 4.8), que por sua vez trata da instalação de todos os pacotes externos no Unity.

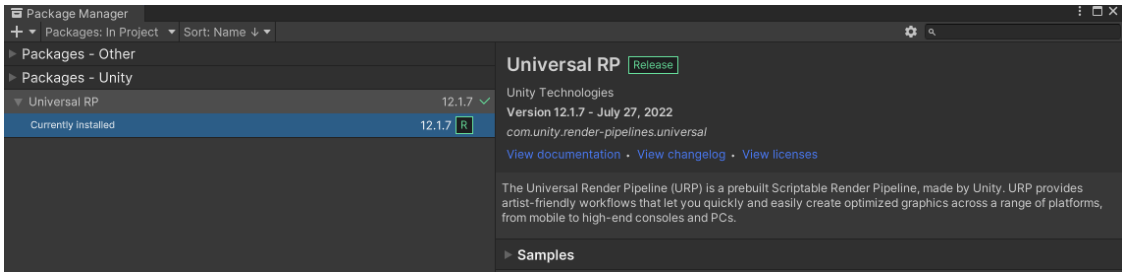


Figura 4.8: Instalação URP

Com a *pipeline* de renderização já instalada, instalou-se alguns pacotes adicionais, de modo a garantir que todas a necessidades do projeto seriam cumpridas. Para tal, mais uma vez, utilizou-se o *Package Manager* e instalaram-se os seguintes pacotes oficiais do Unity:

- ***AI Navigation*** - Contem ferramentas adicionais e recentes para a construção do caminho lógico para a AI.
- ***Post Processing*** - Permite a criação e configuração de pós-processamento de imagem.
- ***ProBuilder*** - Fornece uma nova forma de construção geométrica.
- ***Terrain Tools*** - Expande as ferramentas de construção do próprio Unity, permitindo uma melhor abordagem ao desenvolvimento do terreno.

4.2.2 Primeiro nível - Prisão(Prólogo)

O primeiro nível, de acordo com a narrativa do jogador, funcionará como uma introdução à mesma. Do ponto de vista de jogabilidade, funcionará também como um mapa introdutório às mecânicas base do videojogo. Para tal, como referido no subcapítulo 4.1.3, o primeiro mapa terá sido inspirado nas prisões medievais, sendo que, com isso em mente, deu-se inicio à construção do mesmo.

Construção do terreno

Começou-se por criar um terreno plano, de 50 m por 50 m, e delimitar as áreas jogáveis de acordo com a figura já referida anteriormente (Fig. 4.10). Para delimitar a área jogável, utilizaram-se Assets de estruturas de parede e de grades prisionais medievais, de modo a se enquadrarem no tema do mapa. Adicionou-se também uma textura ao terreno, de modo a simular o chão de uma prisão.

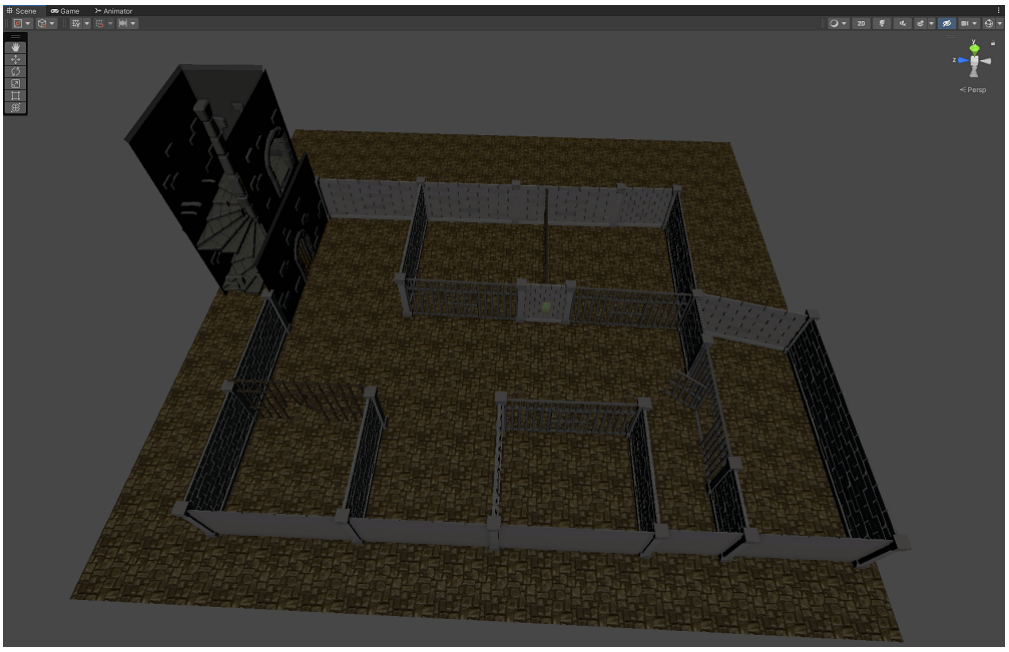


Figura 4.9: Primeiro nível - Prisão(Prólogo) - Terreno

Ao observar-se na (Fig. 4.9), o jogador irá começar no lado direito do mapa, dentro de uma das celas e durante o seu caminho até à saída da prisão, encontra uma cela partida, que será utilizada para introduzir o primeiro ponto de combate, de modo a colocar o jogador à prova.

Com as estruturas do mapa prontas, procedeu-se a alguns testes de jogabilidade para refinar algum tipo de lacuna nas estruturas.

Popular o terreno

Com a estrutura base do mapa desenvolvida, o próximo passo lógico consistiu na alocação de Assets de ambiente, de modo a tornar o mapa mais vivo. Para tal, utilizaram-se Assets de era medieval, como barris, tábuas de madeira, correntes nas paredes, entre outros, de modo a tornar o cenário mais imersivo. Com isto, adicionou-se também algumas *point lights* em tochas nas paredes, já com um efeito de chama preparado, de modo a fornecer uma atmosfera mais convincente, tal como se pode observar na (Fig. 4.10).

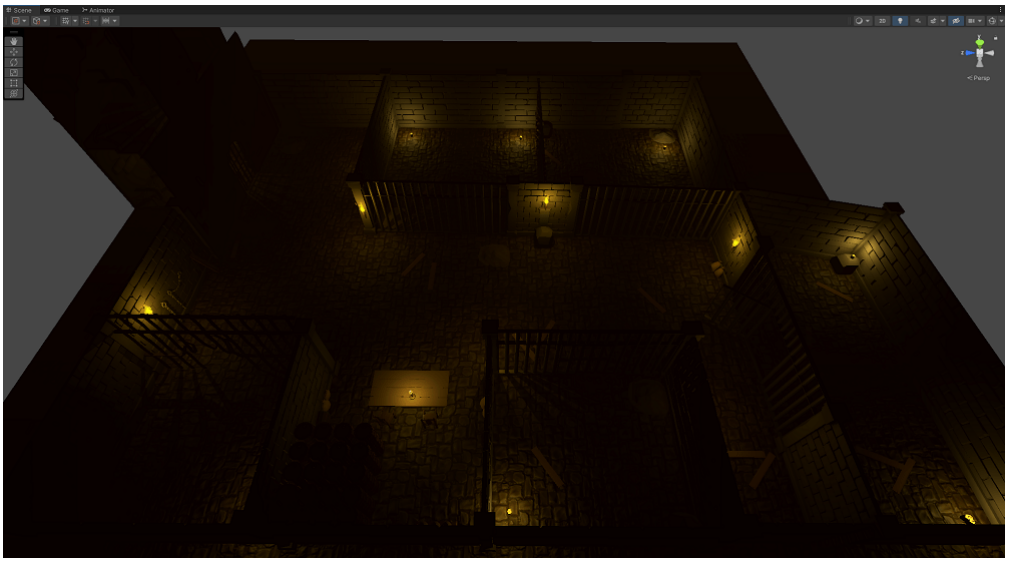


Figura 4.10: Primeiro nível - Prisão(Prólogo) - Terreno populado

Polir o aspeto visual do mapa

Como se pode observar na (Fig. 4.10), a mesma já se parece cada vez mais com uma prisão, mas apesar da quantidade de luzes e de Assets implementados, o ambiente em si encontra-se um pouco monótono, tanto pela falta de cor como pela falta de luz ambiente. Adicionou-se então o pós processamento de imagem, como se pode observar na (Fig. 4.11).

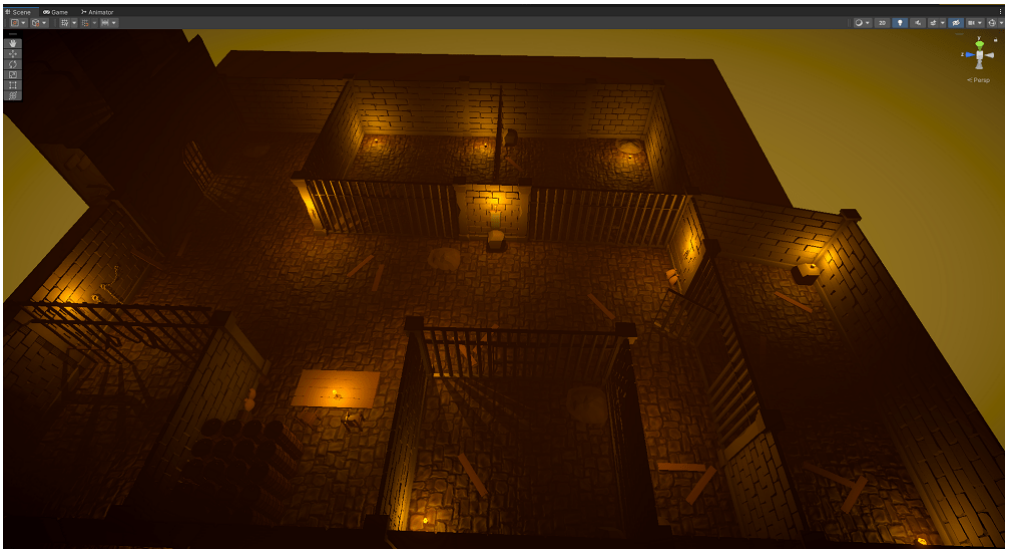


Figura 4.11: Primeiro nível - Prisão(Prólogo) - Com Pós-processamento

Para tal efeito, criou-se um perfil manual de pós processamento de imagem, que forneceu ao mapa um ambiente mais vivo e acolhedor. Adicionou-se também um efeito de nevoeiro que, neste mapa fundamentalmente, serviu para dar um aspeto mais vivo ao ambiente.

Uma vez o mapa concluído, tanto a nível de estrutura como a nível de aspeto visual, como se pode observar na (Fig. 4.12), procedeu-se à colocação de NPCs em cena, de modo a introduzir o primeiro ciclo de jogabilidade.

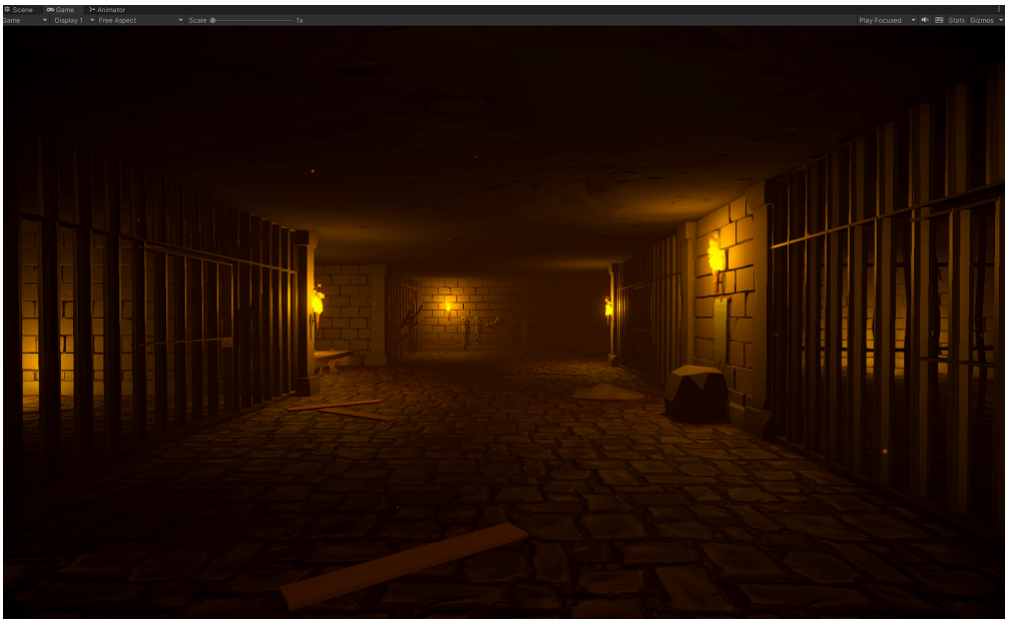


Figura 4.12: Primeiro nível - Prisão(Prólogo) - Vista de jogo

Finalização do mapa

Após a adição de NPCs, verificou-se a estrutura deste nível, com base na jogabilidade, comparando esta à referida na (Fig 4.2). Como se pode observar na (Fig 4.13), a estrutura final do nível, manteve-se fiel ao planeamento inicial, conseguindo-se assim criar um nível bastante simples, acolhedor e que permitirá ao jogador por à prova um pouco das mecânicas base de jogabilidade, antes de progredir para os outros dois níveis.

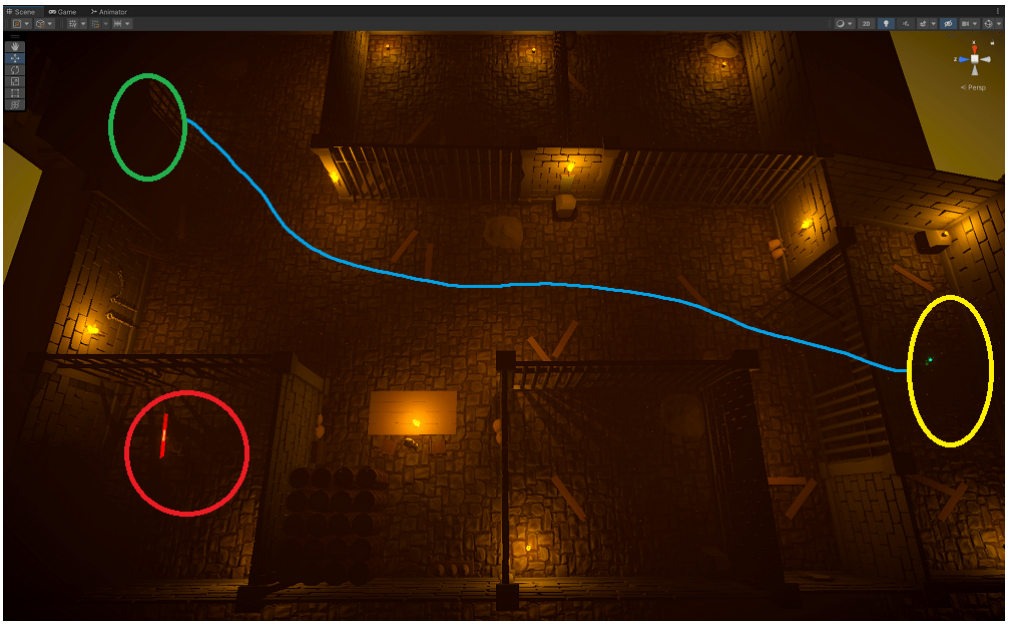


Figura 4.13: Primeiro nível - Prisão(Prólogo) - Estrutura final

4.2.3 Segundo nível - Cidade

O segundo nível, de acordo com a narrativa do jogador, servirá como um nível introdutório ao mundo do videojogo, e que por sua vez, no futuro, atuará como uma "casa"para o mesmo. Como referido no subcapítulo 4.1.3, este mapa terá sido inspirado na série "Made in Abyss"(Fig 4.7). Então, com isso em mente, deu-se inicio à construção do mesmo.

Construção do terreno

O segundo nível, será o primeiro nível em rede, ou seja, não será necessariamente só para um jogador. Por ser uma cidade em volta de uma cratera, possuindo também dimensões bastante maiores do que as do primeiro nível.

Tal como para o primeiro nível, utilizou-se um terreno plano, com 1.5 km por 1.5 km, mas desta vez, com o auxilio do pacote de ferramentas para o terreno, moldou-se o aspeto de uma cratera e poliu-se/texturizou-se o terreno onde a cidade e os pontos de interesse iriam assentar mais tarde. Este pacote de ferramentas utilizado deu acesso a imensos pincéis diversificados, que permitiu esculpir de forma única e mais eficaz.

Como se pode observar na (Fig. 4.14), conseguiu-se moldar uma estrutura de terreno de acordo com a visão inicial, através do uso destes novos pincéis, dando assim inicio ao desenvolvimento da cidade e dos pontos de interesse, tanto dentro da cratera, como em seu redor.

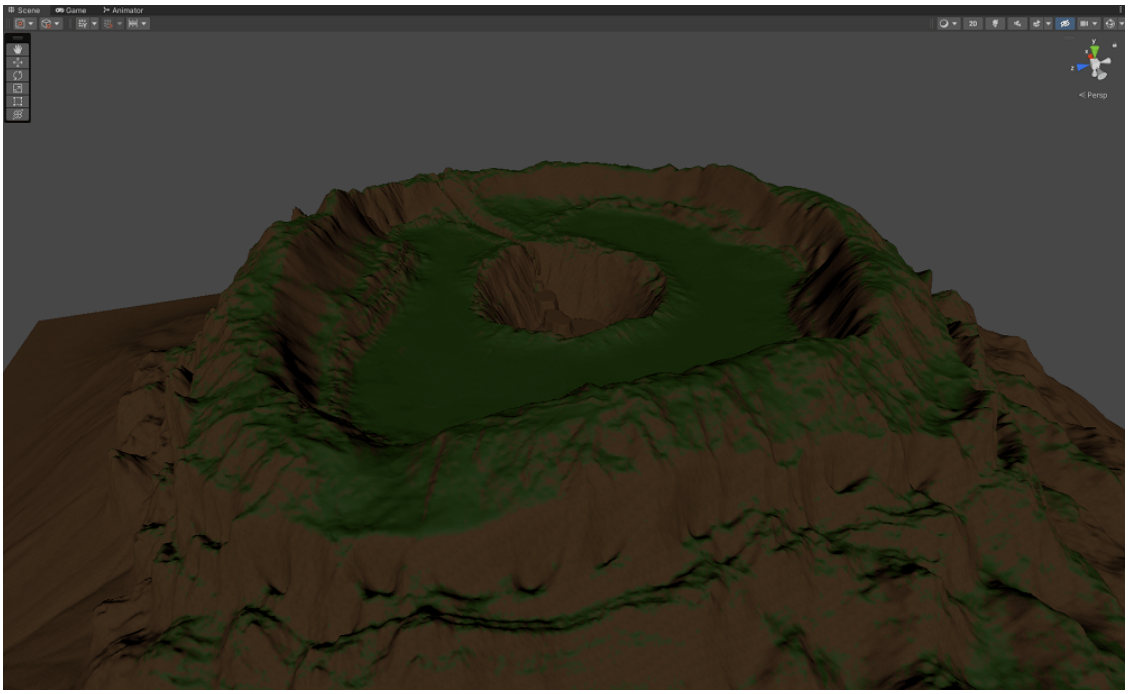


Figura 4.14: Segundo nível - Cidade - Terreno

Construção da cidade

Durante a construção da cidade, teve-se em consideração o facto deste nível possibilitar a funcionalidade de ser jogado em vertente multi jogador. Para tal, construiu-se a cidade com o intuito de ser fácil e cómoda de navegar para um jogador, mas que também conseguisse trazer o mesmo sentimento para quatro jogadores em simultâneo.

Com isto, a cidade foi construida sobre a seguinte estrutura lógica:

- Zona de lazer.
- Zona de mercado.
- Zona de teste ao combate.

Como se pode verificar na (Fig. 4.15), a cidade possui dois centros de venda, denominados por zonas de mercado, assinalados com o circulo amarelo. Dentro destas áreas, o jogador pode vender ou reparar o seu equipamento e encontrará outros NPCs em modo de compra também. Como geralmente, após um melhoramento de arma ou de habilidades, o jogador tende a testar as mesmas, as áreas de teste ao combate (marcadas a vermelho), estão estrategicamente colocadas perto dos mercados, com esse mesmo propósito. O resto do ambiente, possui muita atração visual, visto que o jogador voltará à cidade várias vezes, sendo este, denominado pelas zonas de lazer (marcadas a verde).

Figura 4.15: Segundo nível - Cidade - Estrutura lógica

Construção de pontos de interesse

Os pontos de interesse colocados em redor da cidade, servem tanto para exploração, através da atração visual, como para fornecer missões adicionais para o terceiro nível.

Como se pode observar na (Fig. 4.16), os dois pontos de interesse exploráveis, assinalados a amarelo, encontram-se à saída da cidade, depois da entrada para o terceiro nível, representada pelo circulo verde. O traço azul representa o caminho lógico que o jogador pode tomar, sem restrições, fornecendo ao jogador uma área de exploração mais ampla e por sua vez, um sentido maior de aventura.

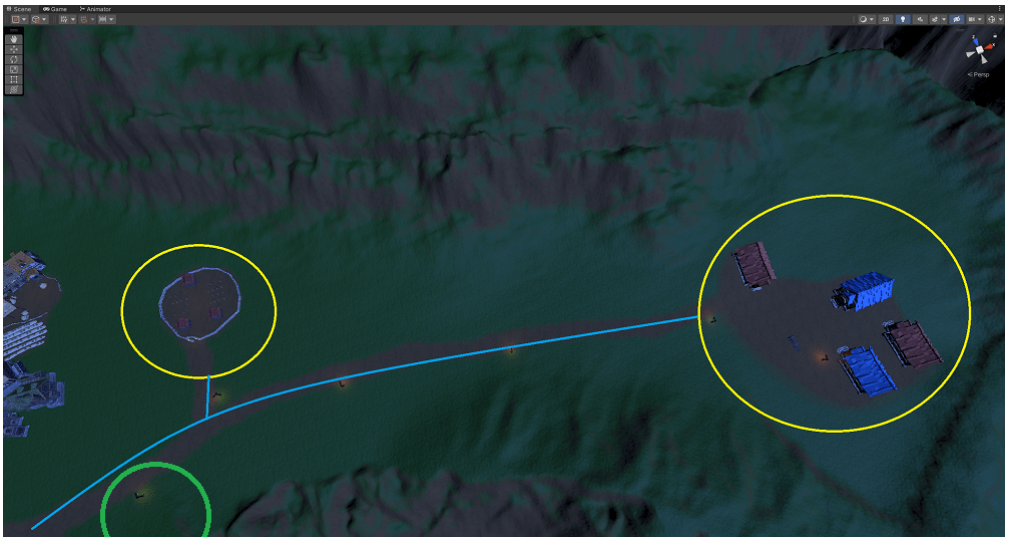


Figura 4.16: Segundo nível - Cidade - Estrutura Pontos de interesse

Popular o ambiente

Uma vez que a cidade e os pontos de interesse já se encontram estabelecidos, deu-se inicio à introdução de flora ambiental, de modo a criar um bioma único em torno da cratera. Para tal, utilizou-se a ferramenta de terreno do Unity, para a colocação de vegetação.

Como se pode observar na (Fig. 4.17), adicionou-se ao mapa áreas florestais, relva e alguns moinhos em redor do topo da cratera, de forma a tornar a cena mais apelativa visualmente. Adicionou-se também uma estrutura retangular com uma *mesh* em movimento, com o objetivo de simular nevoeiro denso, para criar uma atmosfera misteriosa em torno do buraco, impedindo o jogador de ver para além deste.

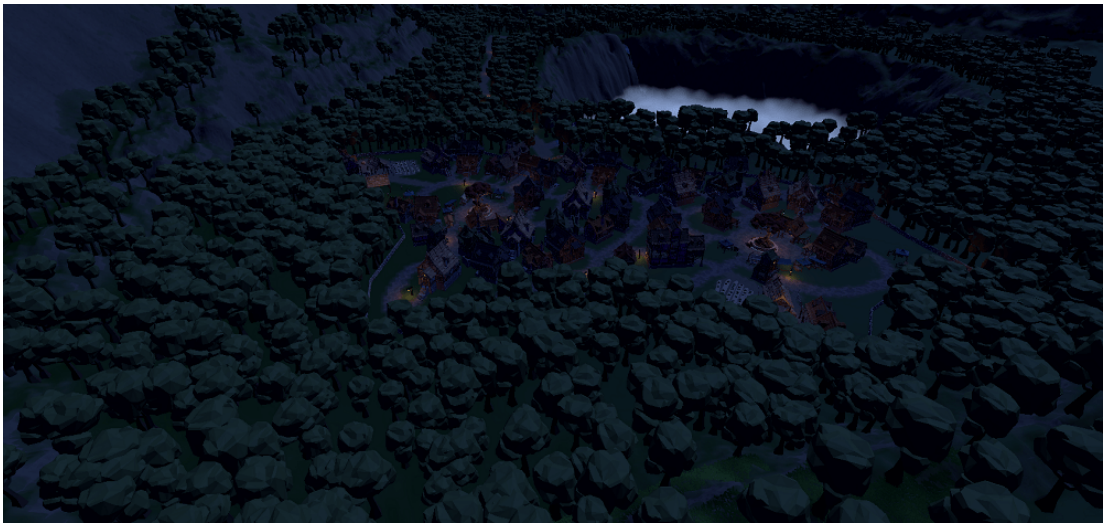


Figura 4.17: Segundo nível - Cidade - Bioma

Popular a cidade

Para o desenvolvimento da cidade utilizou-se, maioritariamente, Assets de edifícios medievais. Mas uma cidade não possui só edifícios. Então, de modo a satisfazer a narrativa do jogador e a estrutura lógica do nível, adicionaram-se bancadas de mercado, distribuídas por dois pontos centrais, postos de reparação de armadura, vários Assets de barris, postes de luz, decorações e edifícios marcados estrategicamente, para que no futuro possam ser modulados interiormente, de modo a tornar a cidade não só mais apelativa, como mais viva.

Após a finalização do ambiente da cidade, seria necessário popular a mesma com NPCs. Para tal, realizou-se uma divisão entre os NPCs, de modo a que uns fossem estáticos, com o intuito de criar ambiente num certo lugar da cidade e outros dinâmicos, que se movem livremente, dentro de parâmetros estabelecidos.

Como se pode observar na (Fig. 4.18), dividiu-se o mapa total em duas vertentes, a vertente da cidade, que inclui NPCs estáticos e dinâmicos (representada pelo circulo amarelo) e a vertente de floresta (representada pelo circulo verde), que também incluirá NPC's, mas distintos dos da cidade.

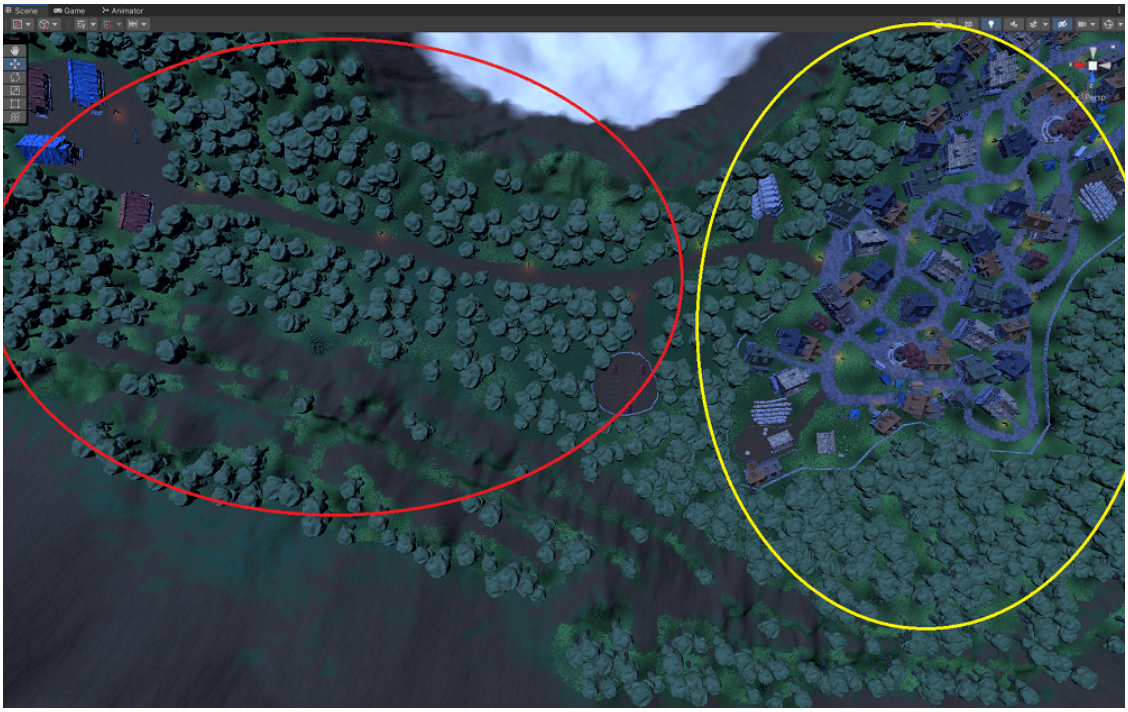


Figura 4.18: Segundo nível - Cidade - Estrutura NPCs

Preparar o mapa para Inteligência Artificial

Após a adição de NPCs em cena, estes necessitam de um caminho lógico que represente a sua área navegável. Por padrão, o Unity não atribui a nenhum terreno a componente de *NavMesh*, sendo esta a essência do movimento da AI. Esta componente trata de calcular todos os pontos geométricos, numa determinada superfície e envia os dados à AI, que através de código próprio, processa o melhor caminho a percorrer.

O mapa da cidade possui muitas ruas, caminhos irregulares e uma vasta área florestal. Para a AI, é fulcral que a mesma só reconheça alguns tipos de caminho como área navegável. Normalmente a *NavMesh* gera uma *mesh* por cima de todo o terreno, mas após a utilização do método referido no subcapítulo 4.2.6, conseguiu-se limitar a *NavMesh* às ruas principais, de modo a criar uma rede navegável dentro dos parâmetros esperados.

Como se pode observar na (Fig. 4.19), a *NavMesh* percorre toda a cidade e floresta, mas restringe o resto do mapa para os NPCs.

Figura 4.19: Segundo nível - Cidade - Estrutura da *NavMesh*

Polir o aspeto visual do mapa

Com o mapa já desenvolvido e testado múltiplas vezes, deu-se então início ao polimento visual da cena. Para tal, semelhante ao primeiro nível, adicionou-se duas componentes, a de nevoeiro e a de pós-processamento de imagem.

Os videojogos que possuem grandes cidades com o objetivo de cativar o jogador a permanecer nelas, possuem todas um detalhe em comum: uma atmosfera muito imersiva. Para obter este efeito, pensou-se no tipo de cores que transmitem uma sensação de "casa". Após várias tentativas e pesquisas, a paleta de cores "*warm*"seria a que mais se adequava, combinando perfeitamente com o estilo *stylized*.

Para atingir tais efeitos, utilizou-se uma combinação de cores, da paleta de "*warm*", em contraste com o resto do ambiente.

Ao observar-se a (Fig. 4.20), consegue-se perceber a importância da adição de pós processamento e o quanto revitalizou aquilo que será a "casa"do jogador, fornecendo assim um ambiente mais rico e vivo.

(a) Cidade sem pós-processamento

(b) Cidade com pós-processamento

Figura 4.20: Segundo nível - Cidade - pós-processamento

4.2.4 Terceiro nível - Gruta

O terceiro nível, de acordo com a narrativa do jogador, é o primeiro nível de entrada para o subsolo e consequentemente o primeiro nível de ação. Como referido no subcapítulo 4.1.2, os biomas deste mapa terão sido inspirados no livro "Viagem ao centro da terra".

Construção do terreno

Tal como para os dois níveis anteriormente referidos, utilizou-se um terreno plano, com 160 m por 160 m, e em conjunto com o pacote de ferramentas para o terreno, moldou-se a base do mapa para a gruta. Como o Unity não possui suporte a desenvolvimento de grutas através das ferramentas de terreno, as mesmas tiveram de ser construidas sob o terreno criado. Para tal, utilizou-se objetos de rocha para moldar o aspeto de uma gruta, clonando a sua semelhança original, de modo a criar três áreas jogáveis diferentes, mas conectadas entre si. A razão por detrás desta estrutura, provem do planeamento, previamente feito para o nível (Fig 4.4).

Ao observar-se a (Fig 4.21), repara-se que a estrutura lógica de construção coincide com a de planeamento, dividindo a gruta em três áreas jogáveis, sendo que as áreas assinaladas a vermelho, representam áreas de combate e de mudança de bioma.

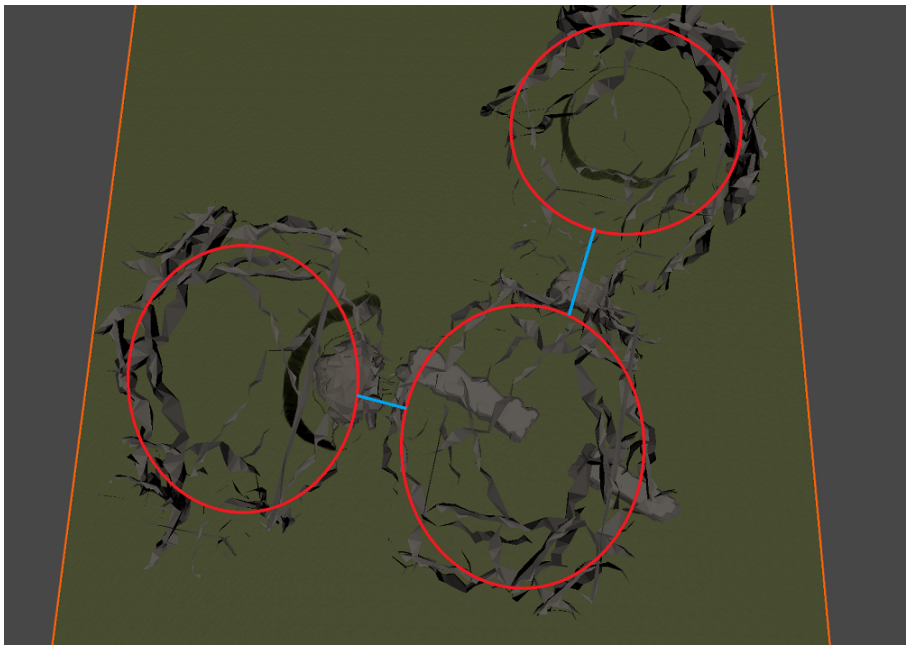


Figura 4.21: Terceiro nível - Gruta - Terreno

Construção da primeira área da gruta

A primeira interação com a gruta, apresenta ao jogador um bioma florestal, combinado com fantasia naval, introduzindo-o aos primeiros aspetos de combate. Para tal, foi necessário desenvolver uma estrutura lógica, não só para guiar o jogador pelo nível, mas como para o incentivar a combater com os inimigos. Com isto em mente, utilizou-se Assets navais, em combinação com alguma flora, de modo a fornecer um aspeto de gruta naval ao jogador.

Ao observar-se a (Fig 4.22), a área de combate (assinalada a vermelho), encontra-se entre o caminho lógico para área seguinte (assinalado a azul) e um ponto de interesse, assinalado a amarelo.

Escolheu-se estrategicamente a posição do ponto de interesse, de modo a despertar a atenção do jogador. Contudo, para este o visitar, o mesmo necessita de derrotar os inimigos que se encontram no caminho. Consequentemente, a posição da área de combate, encontra-se bastante perto da ponte necessária para prosseguir para a próxima área, fazendo com que o jogador tenha de derrotar os inimigos inevitavelmente.

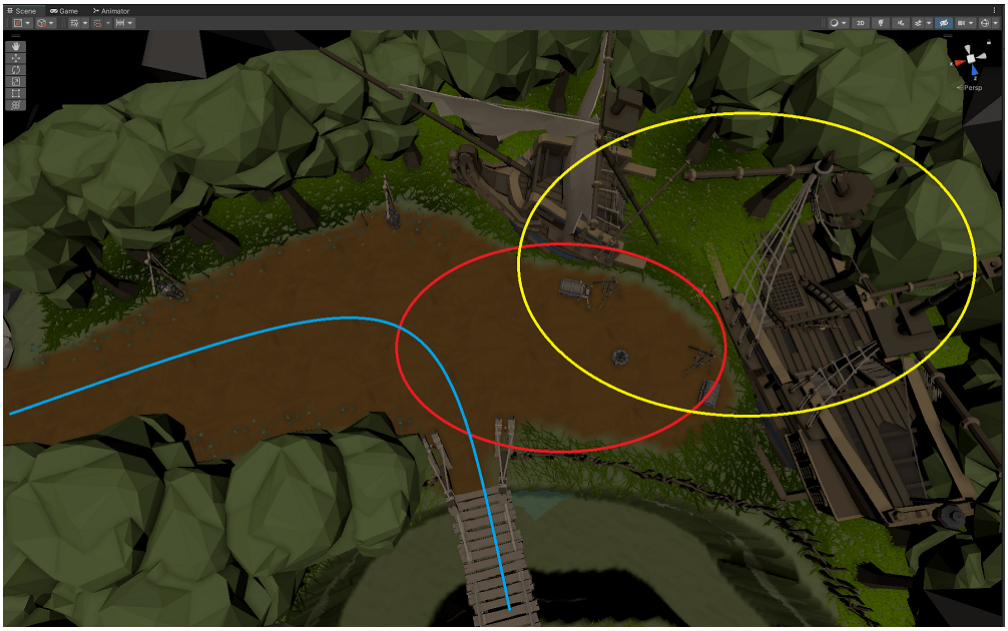


Figura 4.22: Terceiro nível - Gruta - Estrutura Primeira área jogável

Após o jogador ter derrotado todos os inimigos da primeira área, o mesmo consegue então prosseguir para a área seguinte. Em todas as conexões para outras áreas do mapa, existe um ponto de interesse, geralmente destacado através de uma luz mais forte ou de

um objeto mais apelativo, de modo a guiar o jogador pelo nível.

Construção da segunda área da gruta

Uma vez que o jogador tenha conseguido vencer a primeira área, o mesmo pode prosseguir para a segunda, sendo nesta apresentado a um bioma ainda florestal, mas com uma fantasia tribal.

A segunda área jogável terá sido dividida em duas partes. A primeira parte focada em contacto direto com inimigos tribais, de categoria comum e a segunda parte mais focada num bioma florestal mais intenso e com o surgimento do primeiro mini-*boss*. Um mini-*boss*, funciona como um inimigo mais difícil de derrotar, mas não possui as mecânicas de um *boss* normal.

Na construção da primeira parte da segunda área, como referido, utilizou-se Assets tribais, combinando estes com um pouco de flora. Ao observar-se a (Fig 4.23), verifica-se a presença de um ponto de interesse (assinalado a amarelo), que conecta a área anterior a esta, uma zona de combate (assinalada a vermelho) e o caminho lógico do jogador (assinalado a azul). Como se pode analisar, este mapa volta a incentivar o contacto direto com inimigos. A posição destes terá sido escolhida de modo a prevenir que o jogador os ignore, ataque o mini-*boss* da área seguinte e depois acabe encurralado entre os dois.

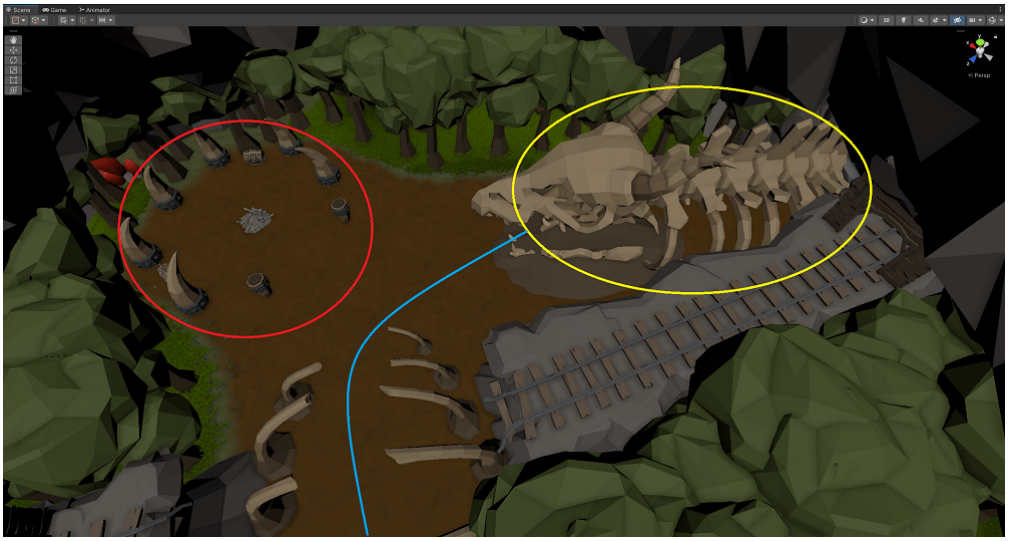


Figura 4.23: Terceiro nível - Gruta - Estrutura segunda área jogável

Uma vez que o jogador ultrapasse os inimigos iniciais, o mesmo entra no pequeno recinto do mini-*boss*, sendo que o jogador é alertado pela mudança de ambiente e a falta de Assets tribais.

Ao observar-se a (Fig 4.24), verifica-se a existência de um ponto de interesse, assinalado a amarelo, que fornece conexão entre esta área e a seguinte, uma área de combate, assinalada a vermelho, e o caminho lógico do jogador, assinalado a azul. Como se pode analisar, a área de combate do mini-*boss* é um pouco maior do que as anteriores, de modo a permitir que o jogador ajuste a sua jogabilidade ao combater o mesmo, uma vez que este dará mais dano que o normal.

Figura 4.24: Terceiro nível - Gruta - Estrutura segunda área jogável

Para o jogador prosseguir para a área seguinte, terá de destruir um portão, perto do ponto de interesse, em que para ser bem sucedido, terá de derrotar primeiro o mini-*boss*, ou este acaba por derrotar o jogador, antes que este consiga destruir o portão completamente.

Construção da área do *boss*

De acordo com o planeamento, no capitulo 4.4, a ultima área jogável deste nível, representa uma área de combate contra um inimigo, mais poderoso que os restantes e com mecânicas adicionais, chamado de *boss*. Esta área possui um bioma único e um estilo mais sombrio, em conjunto com fantasia medieval.

A área de um *boss*, tem que transmitir ao jogador a sensação de perigo e de que algo grande se aproxima. Para tal, existe sempre uma espécie de "*build up*", desde o início da área até à localização do *boss*, de modo a que o jogador se consiga preparar para o que está para vir.

Ao observar-se a (Fig 4.25), repara-se que esta possui dois pontos de interesse, assinalados a amarelo, que fornecem ao jogador informação de algo novo na área. A área possui também um recinto de combate, onde será a luta com o *boss*, assinalada a vermelho e o portão de saída, que se abrirá caso o jogador acabe vitorioso, assinalado a verde. O portão de saída permite que o jogador prossiga para o nível seguinte ou que retorne à cidade.

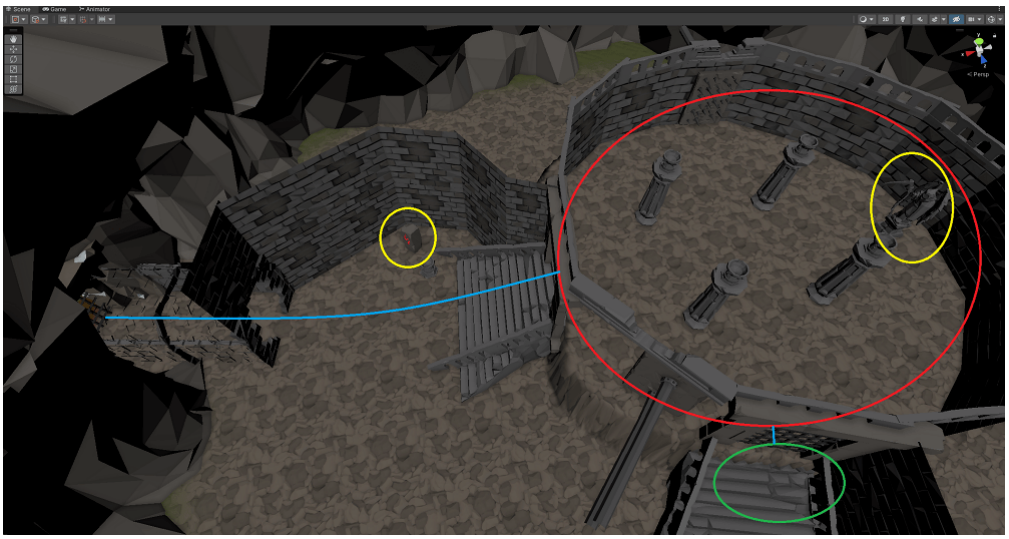


Figura 4.25: Terceiro nível - Gruta - Estrutura área do *boss*

Esta área, ao contrário das anteriores, possui mecânicas adicionais, de modo a enriquecer a batalha com o *boss*, tais como armadilhas laterais, queda de picos no terreno em lugares aleatórios e um ataque frontal, efetuado através da estátua na arena, que cospe fogo horizontalmente, obrigando os jogadores a reposicionar-se atrás dos pilares.

Posicionamento de inimigos

Após a finalização da estrutura do mapa em geral, necessitou-se de popular as áreas de combate. Para tal, como os inimigos são controlados através de AI, criou-se uma *NavMesh*, mas, ao contrário do mapa anterior, como este não possui restrições no caminho navegável, toda a área é jogável, não sendo necessárias grandes modificações.

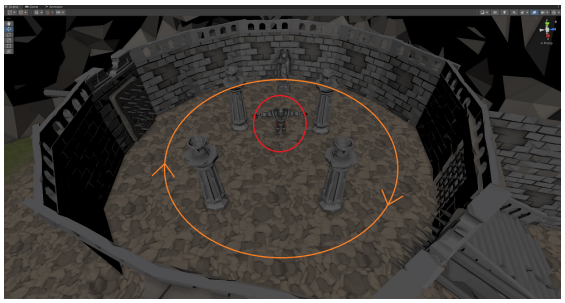
As áreas de combate, à medida que o jogador avança pelo nível, tornam-se progressivamente mais difíceis, de modo a gerar um desafio ao jogador. Se o mesmo jogar em rede, a dificuldade é ajustada de acordo com o número de jogadores em cena. Em todas as áreas de combate (à exceção da área dos "*bosses*"), existe sempre, pelo menos, um inimigo que realiza algum tipo de patrulha. Este tipo de movimentação, acaba por surpreender o jogador, fazendo com que este tenha de rapidamente decidir como enfrentá-los.

Ao observar-se as seguintes figuras (Fig. 4.26), verifica-se que o inimigo, assinalado a vermelho, é o que realiza a patrulha, sendo que os caminhos, assinalados a laranja, representam o seu trajeto. Uma vez que o jogador ataque um inimigo ou entre dentro do seu campo de visão, o mesmo passa para um estado de *chase*, em que persegue o jogador até este o derrotar ou se distanciar o suficiente, fazendo-o perder o foco.

(a) Piratas e humanos

(b) Piratas

(c) Mini-*boss*



(d) *Boss* Final

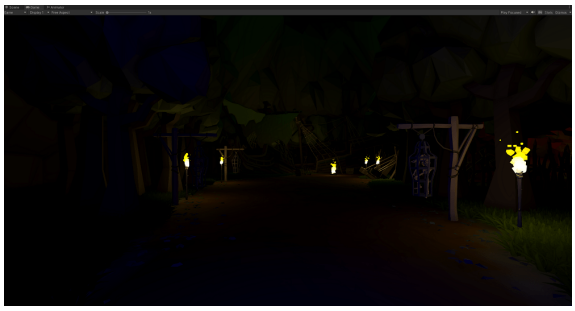
Figura 4.26: Terceiro nível - Gruta - Inimigos

Polir o aspeto visual do mapa

Após o nível ter sido submetido a diversos tipos de testes, estruturais e de jogabilidade, deu-se então início ao polimento visual da cena. Para tal, semelhante aos dois níveis anteriormente referidos, adicionou-se duas componentes essenciais: a de nevoeiro, de modo a criar uma atmosfera mais rica e densa e a componente de pós-processamento de imagem.

Para a componente de cor, a estrutura manteve-se fiel à do segundo nível, tendo sido utilizada a mesma palete de cores *"warm"*, mas com alguns ajustes, de modo a produzir um ambiente semelhante ao do segundo nível, mas com o seu próprio *"twist"*.

Ao observar-se a figura (Fig. 4.26), verifica-se que o cenário possui um ambiente bastante escuro e extremamente aborrecido (Fig:4.27a e Fig:4.27c). No entanto, após a adição das componentes acima referidas, repara-se já num ambiente bastante mais acolhedor (Fig:4.27b e Fig:4.27d) e semelhante ao que encontramos no segundo nível, de modo a fornecer uma sequência lógica de ambiente entre cenas.



(a) Primeira área sem pós-processamento

(b) Primeira área com pós-processamento

(c) Segunda área sem pós-processamento

(d) Segunda área com pós-processamento

Figura 4.27: Terceiro nível - Gruta - pós-processamento

4.2.5 Interface do jogador

Uma vez que o videojogo já se encontrara numa versão jogável, necessitou-se de desenvolver uma interface gráfica básica, de modo a orientar o jogador. Esta interface permite com que o jogador consiga rapidamente verificar aspetos fulcrais da jogabilidade, tais como, o seu nível, as suas habilidades disponíveis, poções coletadas, próximos objetivos, entre outros. Note-se que esta interface serve de complemento para a interface que ficará presente no mapa do nível e não como uma substituição (Exemplo: Símbolos de loja em cima de NPCs, entrada na gruta, entre outros).

Para tal, ao verificar-se a (Fig. 4.28), é possível perceber rapidamente a lógica do posicionamento dos elementos na interface. Para as habilidades, utilizou-se o estilo comum de RPG, em que as mesmas se situam no quadrante central inferior do ecrã, fornecendo assim ao jogador, informação essencial de combate, sem que este tenha de desviar o olhar do objetivo atual.

Para as informações sobre os objetivos, nível, vida e moedas utilizaram-se os extremos do ecrã, tal como é também comum no estilo de RPG, uma vez que não é informação tão vital como a previamente referida, mas é essencial ao ponto de ter de ser colocada no ecrã.

Verifica-se também a presença de uma "mira", que é ativada uma vez que o jogador entre no estado de combate, de modo a permitir que este consiga apontar as suas habilidades e a presença de dígitos no impacto da habilidade, que informam o jogador do total de dano que deferiu ao inimigo.

Figura 4.28: Interface do jogador

4.2.6 Problemas encontrados

Durante a conceção dos mapas, mais precisamente, durante a criação da *NavMesh*, necessitou-se de limitar a mesma a um certo caminho lógico. Tal como referido no capitulo 4.2.3, necessitou-se de limitar a *NavMesh* às ruas da cidade, de modo a que os NPCs não caminhassem por cima de terrenos de relva ou de sítios indesejados.

O Unity não fornece nenhuma opção de recorte de *NavMesh* em URP, sendo que para tal, a forma mais relatada de contornar o problema é de recortar a *mesh* do terreno, com auxilio de um programa externo (exemplo: Blender) e depois combinar ambas as *meshes* no Unity, como dois terrenos diferentes. Desta forma, consegue-se selecionar apenas o terreno cortado e assim construir uma *Navmesh* por cima desse mesmo terreno.

Para a criação da *NavMesh* no segundo mapa, uma vez que recortar a *mesh* se tornava numa decisão final, tendo em conta a arquitetura do nível, decidiu-se recriar este conceito. Para tal, com auxilio de *splines* (ferramenta utilizada para o desenvolvimento de caminhos lógicos sobre um terreno), procedeu-se à criação de um objeto com a mesma formatação que as ruas do nível e através do menu da *NavMesh*, selecionou-se que o único objeto que permitiria um *Walkable path* para os NPCs seria o resultado desta *spline*. Este formato permite ao programador, refazer se necessário o caminho lógico dos NPCs, sem que este tenha a necessidade de alterar novamente a *mesh* do terreno em si, uma vez que se esta já tivesse sido cortada antes, traria agora muito mais complicações.

Com isto, ao observar-se a figura previamente ilustrada (Fig: 4.19), verifica-se o resultado em prática da solução implementada.

4.3 Otimização

Uma vez que os níveis já se encontravam jogáveis, surgiu a necessidade de otimizar os mesmos, de modo a melhorar a qualidade do videojogo e otimizar os seus requisitos.

A otimização é um passo importante durante a etapa de desenvolvimento, pois permite melhorar a alocação de recursos e por consequência, tirar mais partido de outros componentes de *hardware*.

Para efeitos de referência, os requisitos mínimos para a qual o videojogo terá sido otimizado, são os seguintes:

- CPU: Intel Core i5-6200U Dual core
- GPU: Nvidia GTX 940MX 4GB
- RAM: 8GB RAM DDR3
- DISCO: HDD 5200 RPM
- SO: Windows 7 ou superior — Linux

4.3.1 *Deferred Rendering*

Durante o desenvolvimento do videojogo, necessitou-se de tomar uma decisão sobre o tipo de processamento de renderização. Por padrão, o Unity recomenda o uso de *Foward rendering* em sistemas de URP, uma vez que só permite o uso de até oito luzes por objeto, oferecendo assim rápido desempenho numa cena com poucas luzes em tempo real (uma ou menos). No entanto, o projeto necessita de várias luzes, especialmente em tempo real, de modo a fornecerem informação visual para habilidades, sombras dos jogadores, entre outros.

Com isto, optou-se por utilizar uma técnica designada de *Deferred Rendering*, que ao contrário da técnica anterior, esta não possui limite de luzes por objeto, pois esta trata dos cálculos de luz de uma forma mais otimizada. Ou seja, ao invés da quantidade de luzes depender dos objetos em cena, o *deferred rendering* não depende dos mesmos, conseguindo suportar inúmeros pontos de luz, sem grande perda de desempenho.

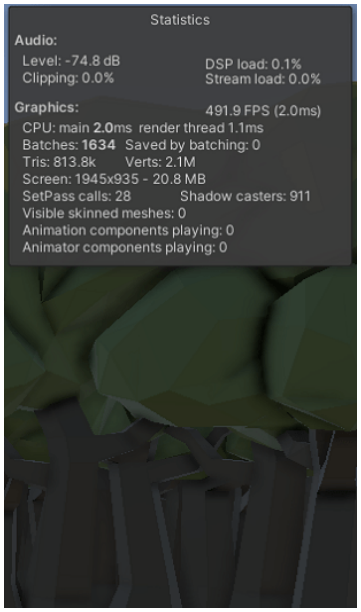
4.3.2 *Static objects e GPU Instancing*

Os mapas do videojogo possuem todos uma característica em comum: o uso intensivo Assets de modo a popularem o ambiente. Para tal, esta quantidade tremenda de objetos necessita de ser toda aglomerada e processada, caso contrário, o processador necessitará de demasiadas chamadas para processar estes.

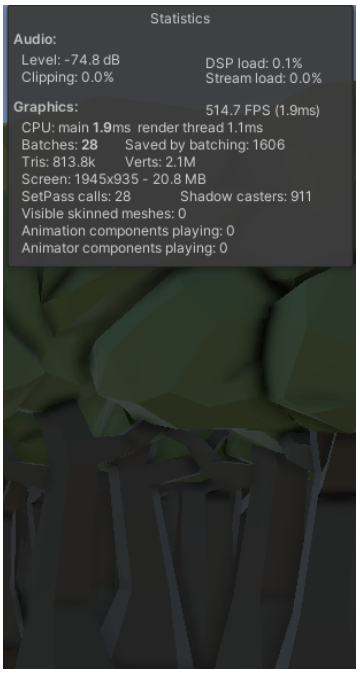
Então, todos os objetos em cena não dinâmicos, estão assinalados como estáticos. Isto permite que o sistema de renderização, uma vez que o jogador execute a cena, aglomere todos os objetos estáticos numa *mesh* própria e processe a mesma em muito menos passagens. Desta forma, reduziu-se crucialmente a quantidade de chamadas ao CPU necessárias.

Por outro lado, para objetos que possuem o mesmo material e a mesma *mesh*(exemplo: árvores, relva, edifícios, entre outros), utilizou-se *GPU Instancing*. Este método permite que o sistema de renderização processe cópias destes objetos, durante a execução da cena, numa única chamada, permitindo assim um número elevado de objetos iguais, sem perda de desempenho.

Observando-se a (Fig 4.29), repara-se que o uso de *GPU Instancing* reduziu bastante a quantidade de *batches* processadas, reduzindo assim o custo de renderização dos objetos em cena.



(a) Sem *GPU Instancing*



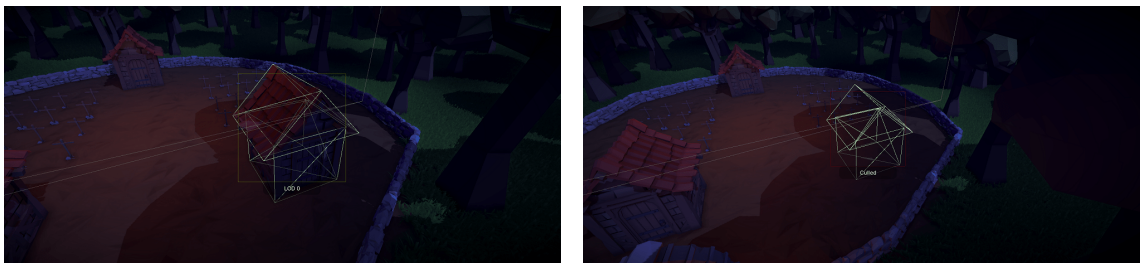
(b) Com *GPU Instancing*

Figura 4.29: *GPU Instancing*

4.3.3 Níveis de detalhe (LOD's - Level of details)

Uma vez que os tipos de *batching* a utilizar já haviam sido definidos, necessitou-se de implementar níveis de detalhe (LOD's) nos objetos, de modo a que este se ajustassem geometricamente, com base na distância da câmera do jogador. Este método permite deformar geometricamente um objeto, mantendo a sua forma original ou até mesmo escondê-lo, reduzindo assim a quantidade de triângulos necessária para o processar, cada vez que este se encontra longe da câmera do jogador.

Ao observar-se a (Fig 4.30), verifica-se que o objeto em questão possui dois níveis de detalhe, sendo o nível normal referente ao que o jogador vê quando se encontra perto do objeto, enquanto que o nível de *culled* acontece quando o jogador se afasta da distância atribuída ao nível de detalhe, sendo este então escondido, de modo a poupar recursos.



(a) Nível de detalhe normal (b) Nível de detalhe *culled*

Figura 4.30: Níveis de detalhe

4.3.4 Otimização da luz/sombras

Bake lighting é um método bastante conhecido de otimização de sombras e de luz, sendo que o mesmo, para funcionar, necessita do uso de luzes *"Baked"* ou *"Mixed"*.

Este método recolhe informação de todos os objetos assinalados como estáticos, em conjunto com todos os tipos de luz acima referidos, processando assim a informação de luz e sombra emitida por estes para uma textura (*Lightmaps*, ex. Fig 2.11). Desta forma os cálculos de luz e sombra em tempo real, deixam de ser processados, sendo que se transformam numa textura.

Durante o desenvolvimento do projeto, decidiu-se utilizar o método de *Bake lighting* com luzes do tipo *"Mixed"*, de modo a poupar recursos e manter fidelidade na quantidade de sombras existentes. Este modo de luz permite também que todos os objetos estáticos

recebam a informação da luz e de sombras uma só vez, ate à realização do *Lightmap*, sendo que depois continuam a produzir luz e sombras para todos os objetos dinâmicos (exemplo: jogadores, NPCs, entre outros), mas descartam os estáticos.

Devido a um erro do Unity, descrito no capitulo 4.3.6, a utilização de luzes *"Mixed"* em combinação com o método de *Bake lighting* não foi permitido, sendo que a otimização de luz acabou por tratada de forma diferente.

Para tal, uma vez que luzes em modo *"Mixed"* já não eram uma opção viável, retornou-se para a utilização de luzes em tempo real. Contudo, muitas luzes em tempo real, implicam muitos cálculos e sombras a ser processadas, levando a um baixo desempenho.

Face a este problema, decidiu-se que só a luz direcional (solar) emitiria sombras e que estas só seriam emitidas de objetos dinâmicos. Para os restantes objetos estáticos, recorreu-se ao auxilio da tecnologia de *Ambient occlusion*, que gera uma sombra nas faces dos objetos que se tocam entre si, de modo a simularem uma sombra, com muito menos custo que uma sombra normal.

Contudo, a utilização de luzes em tempo real, apesar de não emitirem sombras, ainda são luzes que ocupam demasiado tempo de processamento, ao contrário das luzes *"Baked"*, que após a função de *Lightmapping*, se tornam texturas.

Para solucionar o problema em questão, criou-se um *script* (Ilustrado em 4.1), que através da camera do jogador, calcula a distância para os objetos identificados como uma luz (LightF) e se estas estiverem fora da distância atribuída são desativadas, de modo a economizar recursos.

```
public class LightFustrum : MonoBehaviour
{
public float availableDistance;
private float Distance;
private int frames = 0;
public int frametimes = 0;
private GameObject[] Lights;

void Start(){
    Lights = GameObject.FindGameObjectsWithTag("LightF");
}
```

```
void Update () {
        frames++;
        if (frames % frametimes == 0) {
            foreach (GameObject Light in Lights){

                Distance = Vector3.Distance(transform.position,
                    Light.transform.position);
                if (Distance < availableDistance){
                    Light.GetComponent<Light>().enabled = true;
                    }
                if (Distance > availableDistance){
                    Light.GetComponent<Light>().enabled = false;
                    }
                }
        }
}
}
```

Listing 4.1: Script de otimização de luzes

4.3.5 Otimização final

O objetivo geral deste processo de otimização, é de conseguir oferecer ao jogador um ambiente com extrema qualidade e por consequente, elevado desempenho.

Para tal, após a implementação de pequenas medidas extra de otimização, tais como o ajuste da distância de renderização (exemplo: relva, árvores), a distância de sombras, a distância do *Culling* de animações, entre outros, permitiu-se que o videojogo atingisse elevados níveis de desempenho.

Ao analisar-se a (Fig 4.31), verifica-se que após a implementação das medidas de otimização, através do pequeno menu de "*stats*", acedido no canto superior direito da janela de execução do videojogo, a quantidade de chamadas ao CPU (Setpass calls) diminuiu bastante. O numero de *batches* também reduziu crucialmente e por consequência a quantidade de triângulos e de vértices em cena, foi drasticamente reduzida. Todos estes impactaram diretamente para com o desempenho do videojogo, sendo possível verificar-se uma subida notável das imagens por segundo (FPS), mantendo contudo, a mesma fidelidade gráfica.

(a) Sem medidas de otimização

(b) Com medidas de otimização

Figura 4.31: Medidas de otimização

4.3.6 Problemas encontrados

Durante a otimização dos mapas, recorreu-se ao método de *Lightmapping* em conjunto com luzes do tipo *"Mixed"*. Tal como referido no capitulo 4.3.4, foi necessário limitar a quantidade de luzes e sombras em tempo real, de modo a poupar recursos e para tal, o uso deste tipo de luz, em conjunto com *Lightmapping* seria a escolha ideal. Contudo, após várias tentativas e testes, ficou claro que um erro no sistema de *Shader* de terreno do Unity, em combinação com o método de *deferred rendering*, se encontrava presente.

Ao observar-se a (Fig 2.11), verifica-se o resultado esperado desta combinação. No entanto, durante a utilização de *deferred rendering*, o resultado final não produz luz no terreno, tornando-se bastante difícil de visualizar algo em torno do objeto(Fig. 4.32).

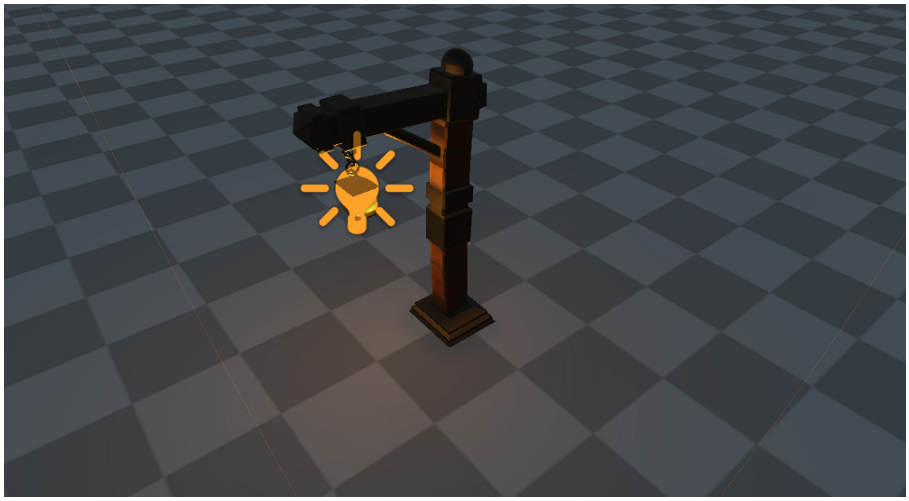
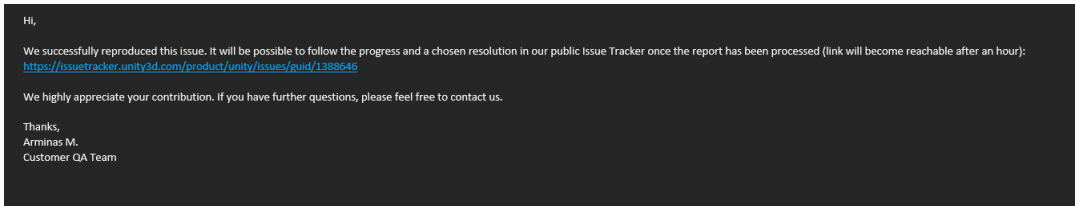


Figura 4.32: Erro reportado Unity

Com isto, submeteu-se um "*bug report*"(Fig. 4.33), que pouco depois acabou por ser declarado como urgente e colocado em fila de espera para ser resolvido/implementado.

À data da escrita deste documento, o *bug*[64], já se encontrava em resolução para as versões beta do Unity 2023, mas ainda em análise e consideração para as versões atuais (2022 e inferiores).


Hi,

We successfully reproduced this issue. It will be possible to follow the progress and a chosen resolution in our public Issue Tracker once the report has been processed (link will become reachable after an hour):
https://issuetracker.unity3d.com/product/unity/issues/guid/1388646

We highly appreciate your contribution. If you have further questions, please feel free to contact us.

Thanks,
Arminas M.
Customer QA Team

Figura 4.33: Ticket do *bug* reportado Unity

Capítulo 5

Testes

Após a etapa de desenvolvimento estar concluída, necessitou-se de testar o videojogo internamente, de modo a garantir que tudo se encontrava estável e pronto para a realização de testes externos.

Para tal, enviou-se o projeto (com um questionário agregado), a vários artistas e entusiastas de videojogos, de forma a obter comentários relevantes, sobre as mais derivadas áreas do mesmo.

5.1 Resultados e discussão

Após a recolha do questionário, conta-se com doze respostas válidas, com respostas individuais direcionadas a todas as vertentes do projeto. Só serão apresentados resultados relevantes para o tema desta dissertação.

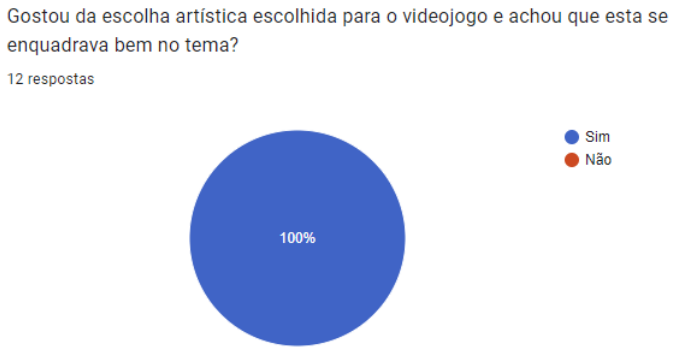


Figura 5.1: Primeira pergunta - Questionário

A primeira pergunta (Fig 5.1), está relacionada com a escolha artística para o videojogo e se esta se enquadrava no tema geral. Repara-se que 100% dos candidatos votaram "Sim", indicando que a combinação de STYLIZED com LOW-POLY terá sido um sucesso.

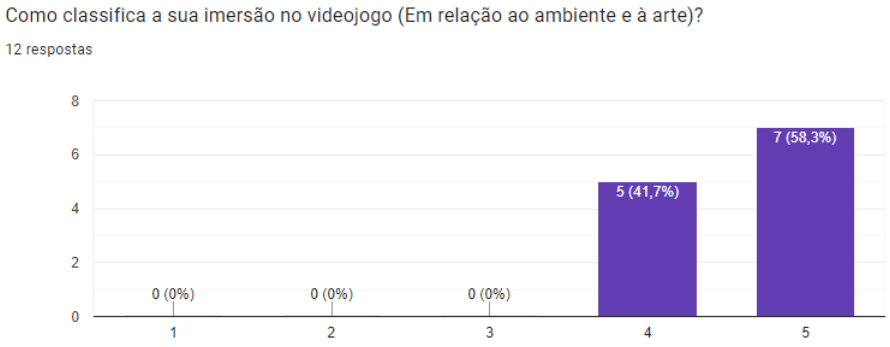


Figura 5.2: Segunda pergunta - Questionário

A segunda pergunta (Fig 5.2), já coloca um pouco de jogabilidade e ambiente em questão, mencionando a imersão do jogador. Tendo uma escala em que 1 = Muito pouco imerso e 5 = Totalmente imerso, os candidatos responderam que o videojogo era totalmente imerso com 58% dos votos e bastante imerso com 42% dos votos restantes. Concluise, com estes resultados, que todo o resultado final do ambiente em junção com o pós-processamento de imagem, foi bem recebido.

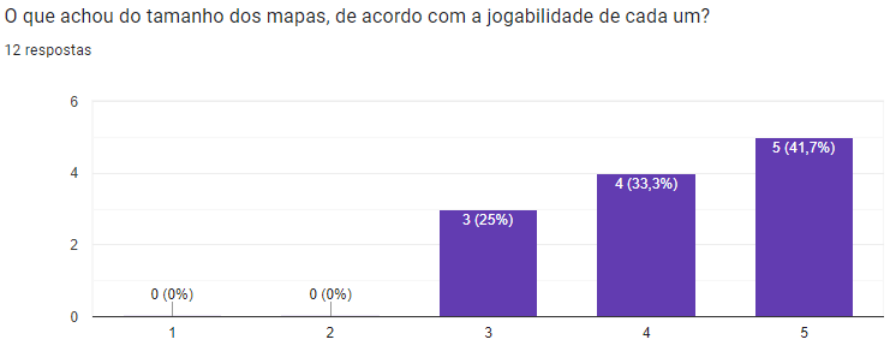


Figura 5.3: Terceira pergunta - Questionário

Para a terceira pergunta (Fig 5.3), relacionou-se o tamanho dos mapas, com a quantidade de jogabilidade/objetivos existentes em cada um. Numa escala em que 1 = Não gostei nada e 5 = Gostei bastante, 42% dos candidatos gostaram bastante do tamanho dos mapas e da quantidade de jogabilidade existente, sendo que os restantes votos que se

encontram entre 3 e 4, demonstram também um voto positivo quanto à questão. Estes resultados demonstram que, em média, os candidatos gostaram do tamanho dos mapas, de acordo com a quantidade de jogabilidade existente.

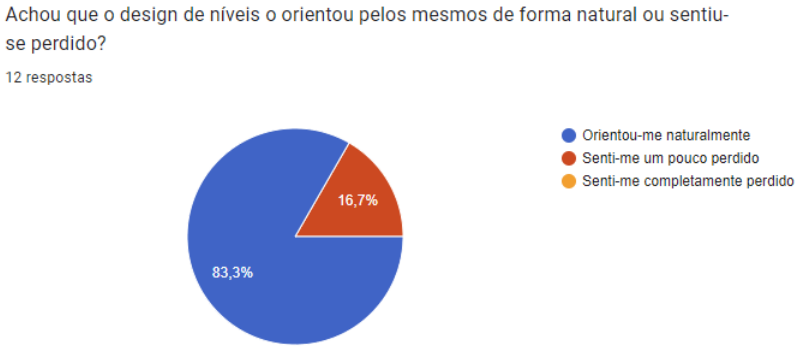


Figura 5.4: Quarta pergunta - Questionário

A quarta pergunta (Fig 5.4) já toca num dos pontos cruciais do *design* de níveis, sendo este o ponto de guiar o jogador intuitivamente, sem que este se sinta perdido. Como se verifica, 83% dos candidatos responderam que os níveis os orientaram intuitivamente e 17% responderam que se sentiram um pouco perdidos, mas após alguma exploração, voltaram ao rumo. Com estes resultados, percebe-se que um dos pontos cruciais do desenvolvimento foi muito bem sucedido.

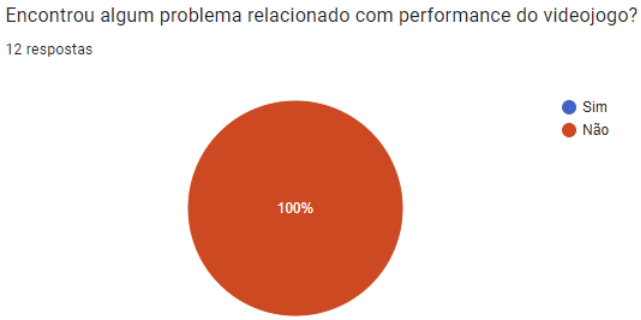


Figura 5.5: Quinta pergunta - Questionário

A quinta pergunta (Fig 5.5) relacionou-se com o desempenho do videojogo nos computadores pessoais dos candidatos. Como se pode observar, nenhum dos candidatos relatou qualquer tipo de erro relacionado com o desempenho. Face a estes valores, verifica-se o quão importante terá sido a etapa de otimização, de modo a prevenir este tipo de problemas.

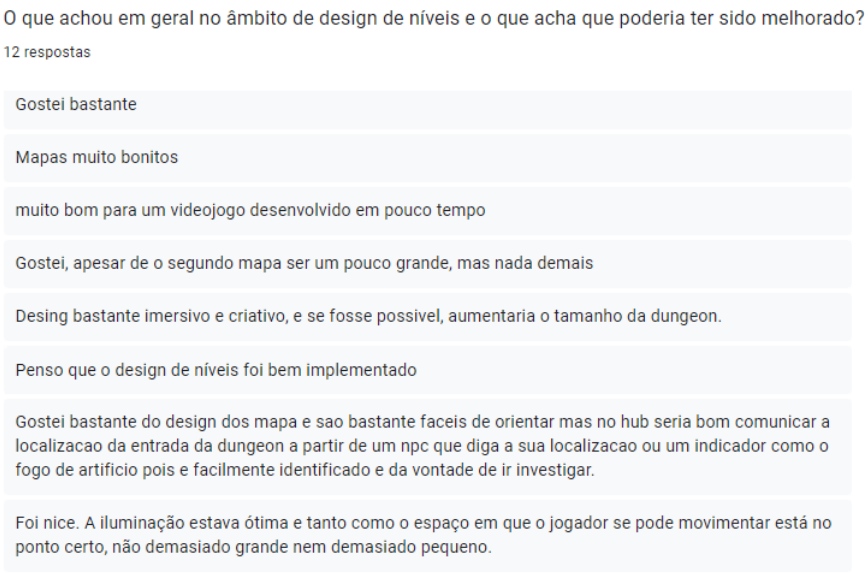
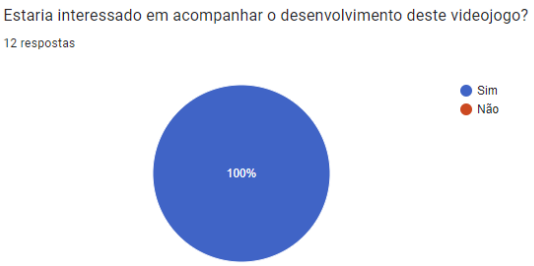


Figura 5.6: Sexta pergunta - Questionário

A sexta e última pergunta relacionada com *design* de níveis (Fig 5.6), pede aos candidatos que forneçam algum tipo de comentário escrito sobre o tema e o que poderia ter sido melhorado de futuro. Ao analisar-se as respostas, verifica-se que são integramente positivas, com alguns pontos extra para trabalho futuro. Com base nestas respostas, considera-se que o *design* de níveis terá sido um sucesso.

(a) Sétima pergunta



(b) Oitava pergunta

Figura 5.7: Sétima e oitava pergunta - Questionário

Por último, realizou-se duas perguntas gerais (Fig 5.7) para perceber o contentamento dos candidatos para com o videojogo e se estes estariam interessados em acompanhar o seu desenvolvimento. Repara-se que a grande maioria dos candidatos, com 84% dos votos, classificou o videojogo como muito bom (valor 4) e repara-se também num interesse elevadíssimo em acompanhar o videojogo, com 100% dos votos. Com base nestes resultados, consegue-se perceber o sucesso desta fase de testes e o valor que este videojogo possui.

Capítulo 6

Conclusão

Perante o levantamento deste estado da arte, conclui-se que o mundo dos videojogos tem vindo a crescer bastante, tanto a nível de lazer, como académico ou profissional.

Consegue-se, também, analisar partes fundamentais para o desenvolvimento de níveis, com especial foco no paradigma multijogador e verificar como alguns pontos são muito importantes, inclusive para o mercado dos videojogos atuais, tendo estes um impacto extremamente positivo caso sejam bem executados.

Durante o desenvolvimento percebe-se o impacto dos métodos discutidos durante todo o estado da arte e como estes podem impactar a jogabilidade/opinião do jogador. Foi necessário analisar várias áreas de videojogos, nomeadamente nas vertentes de RPG e TPS e perceber que padrões de jogabilidade tornavam estas únicas. Com base nestes padrões, desenvolveu-se estruturas lógicas para os mapas, de modo a fornecer um ciclo de jogabilidade lógico ao jogador. Só após estas terem sido estabelecidas, é que se procedeu à construção dos mapas e, por fim, ao polimento visual dos mesmos, de modo a garantir um fidelidade gráfica constante.

Consegue-se, também, perceber que, apesar de muitas vezes descartada, uma boa otimização é chave no desenvolvimento de um videojogo e que sem esta o mesmo pode comprometer a jogabilidade/experiência do jogador.

Após a distribuição de testes, é possível analisar um *feedback* bastante positivo, que indica que o videojogo terá sido bem sucedido, como também o potencial deste para o futuro. Este *feedback* serviu também para melhorar alguns aspetos de *design* de níveis, de modo a permitir que o videojogo expandisse futuramente, com uma qualidade superior.

Capítulo 7

Trabalho futuro

O desenvolvimento deste projeto permitiu aprofundar o conhecimento nas vastas áreas de desenvolvimento de um videojogo. Contudo, este também serve como ponto de partida para outros perceberem a lógica das mesmas, de uma forma mais natural.

O videojogo foi pensado para expandir futuramente. Para tal, uma vez que a estrutura lógica dos níveis ficou bem assente, sendo o segundo nível (cidade) o ponto de retorno do jogador sempre que este termina uma aventura, o videojogo permite uma expansão lógica e rápida de níveis, de modo a permitir uma maior variedade de ambientes/jogabilidade.

Com isto, planeia-se a continuação da gruta, com transição para mundo aberto (ainda dentro da gruta), mas de modo a que cada nível possua um bioma diferente pois, tal como a narrativa indica, esta gruta possuí uma atmosfera única, em que todos os níveis de profundidade, apresentam um tema único.

Por último, pretende-se realizar uma implementação completa de um sistema de *interface* total para o jogador (HUD), de modo a orientar o mesmo pelos níveis de uma forma simples e rápida e fornecer informação clara em mapas de larga escala (exemplo: cidade).

Referências

[1] Gonzalo Frasca. "Videogames of the oppressed". Em: *First person: New media as story, performance, and game* (2004), pp. 85–95.

[2] Mark JP Wolf. *The medium of the video game*. University of Texas Press, 2001. ISBN: 029279150X.

[3] W. Witkowski. *marketwatch*. Obtido de marketwatch, 2020. URL: https://www.marketwatch.com/story/videogames-are-a-bigger-industry-than-sports-and-movies-combined-thanks-to-the-pandemic-11608654990.

[4] Dina Bassiouni e Chris Hackley. "Video Games and Young Children's Evolving Sense of Identity- A Qualitative Study". Em: *Young Consumers* 17 (jul. de 2016). DOI: 10.1108/YC-08-2015-00551.

[5] Dmitri Williams. "Bridging the methodological divide in game research". Em: *Simulation & Gaming* 36.4 (2005), pp. 447–463. DOI: 10.1177/1046878105282275. eprint: https://doi.org/10.1177/1046878105282275. URL: https://doi.org/10.1177/1046878105282275.

[6] S. Egenfeldt-Nielsen. *Overview of research on the educational use of video games*. Nordic Journal of Digital Literacy, 2006. DOI: 10.18261/ISSN1891-943X-2006-03-03.

[7] John Feil e Marc Scattergood. *Beginning game level design*. Thomson Course Technology, 2005. ISBN: 978-1592004348.

[8] Michael Lewis e Jeffrey Jacobson. "Game engines". Em: *Communications of the ACM* 45.1 (2002), p. 27.

[9] John Haas. “A history of the unity game engine”. Em: *Diss. WORCESTER POLYTECHNIC INSTITUTE* (2014).

[10] Unity. *unity. Obtido de unity:* 2021. URL: https://web.archive.org/web/20130312140345/http://docs.unity3d.com/Documentation/Manual/DirectX11.html.

[11] Marie Dealessandri. *What is the best game engine: is Unity right for you?. Obtido de Game industry:* 2020. URL: https://www.gamesindustry.biz/articles/2020-01-16-what-is-the-best-game-engine-is-unity-the-right-game-engine-for-you.

[12] Unity. *unity. Obtido de unity:* 2021. URL: https://store.unity.com/compare-plans.

[13] W. Chyr's. *arstechnica*. Obtido de arstechnica, 2016. URL: https://arstechnica.com/gaming/2016/09/unity-at-10-for-better-or-worse-game-development-has-never-been.

[14] Andrew Sanders. *An introduction to Unreal engine 4*. CRC Press, 2016. ISBN: 9781498765091.

[15] Unreal. *Obtido de unrealengine:* 2021. URL: https://www.unrealengine.com/en-US/faq.

[16] John MacCarthy. “WHAT IS ARTIFICIAL INTELLIGENCE?” Em: *Computer Science Department, Stanford University* (2007). URL: https://ais-lab.di.unimi.it/Teaching/AdvancedIntelligentSystems/Old/IntelligentSystems_2005_2006/Documents/Symbolic/04_McCarthy_whatisai.pdf.

[17] Bernard Marr. *What Is The Impact Of Artificial Intelligence (AI) On Society?. Obtido de Bernard Marr:* 2021. URL: https://bernardmarr.com/what-is-the-impact-of-artificial-intelligence-ai-on-society/#:~:text=Artificial%5C%20intelligence%5C%20can%5C%20dramatically%5C%20improve,creativity%5C%20and%5C%20empathy%5C%20among%5C%20others..

[18] R. Barrera. *Unity 2017 Game AI Programming - Third Edition: Leverage the power of Artificial Intelligence to program smart entities for your games, 3rd Edition*. Packt

Publishing, 2018. ISBN: 9781788393294. URL: https://books.google.pt/books?id=qtRJDwAAQBAJ.

[19] Ben Goertzel e Cassio Pennachin. *Artificial general intelligence*. Vol. 2. Springer, 2007. ISBN: 978-3-540-68677-4.

[20] Roman V Yampolskiy. *Artificial superintelligence: a futuristic approach*. cRc Press, 2015. ISBN: 978-1482234435.

[21] Call of Duty. *Call of Duty. Obtido de Call of Duty:* URL: https://www.callofduty.com/pt.

[22] David Vonderhaar e Andy Hartup. *So, how do you make the perfect Call of Duty multiplayer map? Obtido de Games radar:* 2015. URL: https://www.gamesradar.com/perfect-call-duty-multiplayer-map/.

[23] Andrzej Marczewski. *Gamification User Journey Framework. Obtido de Gamified UK:* 2017. URL: https://www.gamified.uk/2017/03/13/gamification-user-journey-framework/.

[24] Mirror's Edge. *Mirror's Edge. Obtido de EA:* 2009. URL: https://www.ea.com/en-gb/games/mirrors-edge/mirrors-edge.

[25] Bioshock. *Bioshock. Obtido de Steam:* 2007. URL: https://store.steampowered.com/app/7670/BioShock/.

[26] Computer History Museum. *CHM Revolutionaries: Game Changers- Mark Cerny with EA's Rich Hilleman*. Youtube. 2012. URL: https://www.youtube.com/watch?v=DiboVZsXYXY.

[27] Playstation. *Ratchet & Clank was the best-selling game on PlayStation Store last month. Obtido de Playstation blog:* 2016. URL: https://blog.playstation.com/archive/2016/05/11/ratchet-clank-was-the-best-selling-game-on-playstation-store-last-month/.

[28] Ratchet&Clank. *Ratchet&Clank. Obtido de Playstation:* 2016. URL: https://store.playstation.com/pt-pt/product/EP9000-CUSA01073_00-RCPS400000000000.

[29] Assassin's Creed. *Assassin's Creed. Obtido de Ubisoft:* URL: https://www.ubisoft.com/en-us/game/assassins-creed/all-games.

[30] Alexandre Mandryka. *Fun and uncertainty. Obtido de Game Whispering:* 2012. URL: http://gamewhispering.com/fun-and-uncertainty/.

[31] Raph Koster. *A Theory of Fun for Game Design.* O'Reilly Media, Inc., 2013. ISBN: 978-1449363215.

[32] Splinter's Cell. *Splinter's Cell. Obtido de Ubisoft:* URL: https://store.ubi.com/uk/game?pid=58eb8edd88a7e338398b4567&dwvar_58eb8edd88a7e338398b4567_Platform=pcdl&edition=Splinter%5C%20Cell%5C%20Collection&source=detail.

[33] Pascal Luban. *Multiplayer Level Design In-Depth. Obtido de Game developer:* 2006. URL: https://www.gamedeveloper.com/design/multiplayer-level-design-in-depth-part-1-the-specific-constraints-of-multiplayer-level-design.

[34] Wikipedia. *Co-op. Obtido de Wikipedia:* 2021. URL: https://en.wikipedia.org/w/index.php?title=Cooperative_video_game&oldid=1062827772.

[35] Assassin's Creed Unity. *Assassin's Creed Unity. Obtido de Ubisoft:* 2014. URL: https://www.ubisoft.com/en-gb/game/assassins-creed/unity.

[36] Wikipedia. *Online games. Obtido de Wikipedia:* 2021. URL: https://en.wikipedia.org/w/index.php?title=Online_game&oldid=1061736463.

[37] Benjamin Bauer. "Ben's small bible of realistic multiplayer leveldesign". Em: (2004). URL: http://www.benb-design.net/Articles/benb_article02.pdf.

[38] Pluralsight. *Light Up Your World: How Lighting Makes All the Difference for Games. Obtido de Pluralsight:* 2014. URL: https://www.pluralsight.com/blog/film-games/understanding-the-importance-of-lighting-for-games.

[39] Rockstar Games. *Red Dead Redemption 2. Obtido de Rockstargames:* URL: https://www.rockstargames.com/reddeadredemption2/restricted-content/agegate/form?redirect=https%5C%3A%5C%2F%5C%2Fwww.rockstargames.com%5C%2Freddeadredemption2%5C%2F&options=&locale=en_us.

[40] Luke Reilly. *Red Dead Redemption 2 Review. Obtido de IGN:* 2020. URL: https://www.ign.com/articles/2018/10/25/red-dead-redemption-2-review.

[41] Unreal. *Types of lights. Obtido de Unreal:* 2021. URL: https://docs.unrealengine.com/4.27/en-US/BuildingWorlds/LightingAndShadows/LightTypes/.

[42] Unity Documentation. *Types of light. Obtido de Unity Documentation:* 2020. URL: https://docs.unity3d.com/Manual/Lighting.html.

[43] Unity Documentation. *Textures. Obtido de Unity Documentation:* 2020. URL: https://docs.unity3d.com/Manual/Textures.html.

[44] Unity Documentation. *Textures types. Obtido de Unity Documentation:* 2020. URL: https://docs.unity3d.com/Manual/TextureTypes.html.

[45] Cyberpunk 2077. *Cyberpunk 2077. Obtido de Cyberpunk:* 2020. URL: https://www.cyberpunk.net/pt/en/.

[46] The guardian. *Cyberpunk 2077: how 2020's biggest video game launch turned into a shambles. Obtido de The Guardian:* 2020. URL: https://www.theguardian.com/games/2020/dec/18/cyberpunk-2077-how-2020s-biggest-video-game-launch-turned-into-a-shambles.

[47] Metacritic. *Cyberpunk 2077. Obtido de Cyberpunk:* 2020. URL: https://www.metacritic.com/game/pc/cyberpunk-2077.

[48] Ben Garney Eric Preisz. *Video Game Optimization.* Cengage Learning PTR, 2010. ISBN: 978-1598634358.

[49] S. Santos. *What Is A Good FPS For Gaming?. Obtido de Blue Cine Tech:* 2020. URL: https://www.bluecinetech.co.uk/what-is-a-good-fps-for-gaming/.

[50] God of War. *God of War PC. Obtido de Steam:* 2021. URL: https://store.steampowered.com/app/1593500/God_of_War/.

[51] Metacritic. *God of War PC. Obtido de Metacritic:* 2021. URL: https://www.metacritic.com/game/pc/god-of-war.

[52] Eddie Makuch. *God Of War For PC Is Doing Great, Reaches No. 1 On Steam. Obtido de Gamespot:* 2021. URL: https://www.gamespot.com/articles/god-of-war-for-pc-is-doing-great-reaches-no-1-on-steam/1100-6499698/.

[53] Unity Documentation. *Occlusion culling. Obtido de Unity Documentation:* 2020. URL: https://docs.unity3d.com/550/Documentation/Manual/OcclusionCulling.html.

[54] Tino. *Lightmapping in Unity 5. Obtido de Sassybot:* 2015. URL: https://sassybot.com/blog/lightmapping-in-unity-5/.

[55] Unity Documentation. *The Inspector window. Obtido de Unity Documentation:* 2020. URL: https://docs.unity3d.com/Manual/UsingTheInspector.html.

[56] Unity Documentation. *Mesh. Obtido de Unity Documentation:* 2020. URL: https://docs.unity3d.com/ScriptReference/Mesh.html.

[57] Linda Candy e Ernest Edmonds. "Practice-Based Research in the Creative Arts: Foundations and Futures from the Front Line". Em: *Leonardo* 51 (fev. de 2018), pp. 63–69. DOI: 10.1162/LEON_a_01471.

[58] 3DCoat. *Princípios Básicos Da Modelagem Low-Poly. Obtido de 3DCoat:* URL: https://3dcoat.com/pt/articles/article/basic-principles-of-low-poly-modeling/.

[59] Blizzard Entertainment. *UNITING GAMEPLAY AND STYLE: BEHIND OVERWATCH 2'S COMPLEX MAP DESIGN. Obtido de playoverwatch:* 2022. URL: https://playoverwatch.com/en-us/news/23785339/uniting-gameplay-and-style-behind-overwatch-2-s-complex-map-design/.

[60] Wikipedia. *Made in Abyss. Obtido de Wikipedia:* 2022. URL: https://en.wikipedia.org/w/index.php?title=Made_in_Abyss&oldid=1110298782.

[61] Made in Abyss Wiki. *The Abyss. Obtido de Made in Abyss Wiki:* URL: https://madeinabyss.fandom.com/wiki/The_Abyss.

[62] Jules Verne. *Journey to the Center of the Earth*. RHUS, 1991. ISBN: 0553213970.

[63] Synty Studios. *Obtido de SyntyStudios:* URL: https://www.syntystudios.com.

[64] Unity Issue Tracker. *Obtido de Unity:* URL: https://issuetracker.unity3d.com/issues/terrain-slash-urp-mixed-light-baking-with-shadowmask-on-urp-with-terrainlit-shader-doesnt-work-as-intended-when-using-deferred-rendering.