UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
CENTRO DE LETRAS E ARTES
ESCOLA DE BELAS ARTES
DEPARTAMENTO DE COMUNICAÇÃO VISUAL – DESIGN

ALEXANDER EPAMINONDAS DE CARVALHO

LEVEL DESIGN DE CENÁRIO PARA JOGO *MULTIPLAYER*

RIO DE JANEIRO
2021

Alexander Epaminondas de Carvalho

Level design de cenário para jogo *multiplayer*

Trabalho de Conclusão de Curso apresentado à Escola de Belas Artes da Universidade Federal do Rio de Janeiro, como requisito necessário à obtenção do grau de bacharel em Comunicação Visual – Design.

Orientador: Prof. Dr. Carlos de Azambuja Rodrigues.

Rio de Janeiro
2021

CIP - Catalogação na Publicação

CC331l

Carvalho, Alexander Epaminondas de
Level design de cenário para jogo multiplayer / Alexander Epaminondas de Carvalho. -- Rio de Janeiro, 2021.
44 f.

Orientador: Carlos de Azambuja Rodrigues.
Trabalho de conclusão de curso (graduação) - Universidade Federal do Rio de Janeiro, Escola de Belas Artes, Bacharel em Comunicação Visual Design, 2021.

1. game design. 2. level design. 3. jogabilidade. 4. experiências do jogador. I. Rodrigues, Carlos de Azambuja, orient. II. Título.

Elaborado pelo Sistema de Geração Automática da UFRJ com os dados fornecidos pelo(a) autor(a), sob a responsabilidade de Miguel Romeu Amorim Neto - CRB-7/6283.

Alexander Epaminondas de Carvalho

LEVEL DESIGN DE CENÁRIO PARA JOGO *MULTIPLAYER*

Trabalho de Conclusão de Curso apresentado à Escola de Belas Artes da Universidade Federal do Rio de Janeiro, como requisito necessário à obtenção do grau de bacharel em Comunicação Visual – Design.

Aprovado em: ___/___/____

Banca Examinadora

_____

Prof. Dr. Carlos de Azambuja Rodrigues
(Orientador – UFRJ)

_____

Profª. Dra. Elizabeth Motta Jacob
(Membro – UFRJ)

_____

Profª. Dra. Fabiana Oliveira Heinrich
(Membro – UFRJ)

AGRADECIMENTOS

Agradeço à minha família, por todo o apoio e compreensão, contribuindo para que eu pudesse prosseguir com o TCC.

Agradeço ao meu orientador, Carlos Azambuja, pelos conselhos e oportunidades que me ofereceu ao longo deste processo.

Agradeço, também, aos meus professores da UFRJ, pelos conhecimentos que ajudaram em minha formação.

RESUMO

CARVALHO, Alexander Epaminondas de. ***Level design* de cenário para jogo *multiplayer***. 2021. 44f. Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Rio de Janeiro, Escola de Belas Artes, Bacharel em Comunicação Visual Design, 2021.

Este Trabalho de Conclusão de Curso consiste em refletir acerca de todas as etapas que o *level designer* deve percorrer e as escolhas que ele deve considerar para garantir uma excelente experiência ao jogador de *videogame*. A área de *game design*, que trata do desenvolvimento das regras de um jogo, é o ponto de partida deste trabalho. O *CS:GO* é utilizado, então, como base de análise de interação entre as áreas de *game design* e *level design*. Foi criado um nível de cenário, tendo o CS:GO como referência, para exemplificar essa discussão sobre a atuação do *level designer*.

Palavras-chave: *game design*, *level design*, jogabilidade, experiências do jogador.

LISTA DE FIGURAS

SUMÁRIO

1 **INTRODUÇÃO**

A adequação do jogo ao público-alvo é uma das muitas preocupações que os *designers* têm ao desenvolver um novo *game*. Schell (2015) destaca que as dificuldades já começam na escolha da ideia do jogo, pois muitos se perdem ou no excesso de ideias ou na falta de qualidade delas; mas, por fim, algo é escolhido, embora não seja possível saber de antemão se a ideia é boa.

O autor defende que o projeto de design deve ser submetido ao que ele chama de "os oito filtros". Basicamente os filtros levam o *designer* à reflexão sobre a definição do público-alvo, se o jogo é inovador, se é realizável em termos tecnológicos, se é comercializável etc e defende a testagem do jogo para saber se ele é divertido.

O filtro nº 3 está relacionado ao design de experiência. O conselho é: "para aplicar esse filtro, leve em consideração tudo o que você sabe sobre como criar uma boa experiência, incluindo estética, curvas de interesse, tema ressonante e equilíbrio do jogo." (SCHELL, 2015, p. 91, tradução nossa)

Os caminhos percorridos neste projeto têm o objetivo principal de demonstrar os procedimentos cabíveis ao *level design* no desenvolvimento de jogos, em meio às demais áreas que se inter-relacionam com esse setor durante todo o processo.

No capítulo "o que é *game design*?", procuramos trabalhar os conceitos dessa área a partir do exemplo de um jogo muito conhecido mundialmente: o *Counter-Strike*.

Já no capítulo "o que é *level design*?", demonstramos quais são as principais funções do *level design*, como ele se diferencia e, ao mesmo tempo, tem uma relação de codependência com o game design.

E, no capítulo "*level design* aplicado no *CS:GO*", as discussões realizadas anteriormente evoluem para a parte prática, permitindo demonstrar detalhadamente a atuação de um *level designer*.

2 **O QUE É *GAME DESIGN*?**

Existem várias interpretações do que seria *game design*. Algumas definições vêm do meio acadêmico e muitas outras surgiram da experiência de *game designers*, que obtiveram suas definições através da prática. Mas nenhuma delas convergem em um único ponto, dificultando a criação de uma definição objetiva sobre o que é *game design*.

O mais importante sobre esse assunto não é ter uma definição clara e universal, porque muitas dessas definições funcionam se forem aplicadas isoladamente em casos específicos. Saber a função do *game design* é o ponto mais importante. E essa função está relacionada diretamente com os objetivos externos do jogo. Esses objetivos têm como ponto central atrair a atenção e interesse do jogador. É por meio desses objetivos externos que sabemos qual será a principal função do *game design* e os objetivos que pretendem alcançar.

Por esse motivo, para definir o seu papel dentro do desenvolvimento de um jogo, temos que buscar entender qual é a sua função principal, o seu objetivo e o que tentar alcançar com essas duas coisas. Esses tópicos serão discutidos mais adiante, tendo o *Counter-Strike* (*CS*) como modelo de análise.

O *Counter-Strike* – ou *CS:GO*, como é mais conhecido pelo público – foi lançado em 1999, por Minh Le e Jess A. Cliffe, e tem sua origem em um *Mod*[1] (*modification*) de *Half-life 2*. Ele é um jogo *multiplayer* de tiro em primeira pessoa. Como destaca Rabin (2011, p. 33), "a maioria dos *videogames* pode ser relacionada a um gênero particular ou classificada como híbrida de dois ou mais gêneros. Esses gêneros foram aparecendo durante os anos, em geral como resultado de tentativas e erros, mas frequentemente também como uma evolução." O gênero "tiro em primeira pessoa" se caracteriza, por sua vez, pelo fato de permitir que o jogador tenha a perspectiva de visão da personagem; além de ser, também, um jogo de ação.

Após ser adquirido pela *Valve* – mesma desenvolvedora de *Half-Life* –, o *Counter-Strike* passou por várias adaptações até chegar na versão que é jogada hoje por vários jogadores

[1] É uma abreviatura da palavra em inglês *modification.*

casuais e profissionais. Ele é considerado um dos jogos pioneiros no seguimento dos *e-sports*[2], promovendo diversos campeonatos regionais e mundiais.

O *CS:GO* oferece vários modos de jogabilidade e cada um deles tem suas características intrínsecas. Por esse motivo, cada modo de jogo tem as suas próprias regras. Elas ditam a forma como o jogador precisa jogar para conseguir o objetivo do jogo. Essas regras também servem para limitar o que o jogador pode fazer durante uma partida.

Existem sete modos de jogo no *Counter-Strike* para experimentar, apenas um deles será utilizado, neste trabalho, como base para a análise de *game design*. Trata-se do modo competitivo do *CS:GO.*

2.1 OBJETIVOS EXTERNOS DO *GAME DESIGN* NO *CS:GO*

Ao analisarmos o jogo, podemos destacar dois grandes objetivos externos que ele tem a intenção de alcançar. São os seguintes objetivos: oferecer uma experiência de combate entre equipes e ter uma experiência de trabalho em equipe.

2.1.1 **O combate entre equipes**

Uma das principais experiências e mecânicas de jogabilidade que o *Counter-Strike* oferece ao jogador é o combate entre equipes. Por ser um jogo muito difícil de jogar sozinho, o *CS:GO* acaba se definindo, naturalmente, como um jogo de equipe. É muito mais recompensador montar uma estratégia com seus aliados e vencer uma rodada do que sair sozinho contra cinco adversários.

O combate entre equipes define o *Counter-Strike* de tal forma que fica quase impossível jogar sozinho. Mesmo assim, às vezes, o jogador não tem alternativas: a sua equipe pode ser

[2] Termo utilizado para definir modalidade esportiva digital.

completamente neutralizada e poderá restar apenas um único integrante para cumprir o objetivo do time.

Isso faz parte das circunstâncias da jogabilidade, porém o trabalho em equipe não se justifica apenas por causa da dificuldade. A questão é que um dos objetivos principais do jogo é, simplesmente, ter uma experiência de combate entre equipes.

2.1.2 **O trabalho em equipe**

Ter uma boa estratégia, se comunicar com seus aliados, comprar equipamentos para um integrante da equipe, informar posições do oponente e seguir o que foi planejado são alguns dos diversos tipos de ações disponíveis para o jogador experimentar. Toda essa interação entre os jogadores se demonstra positiva e recompensadora para eles, mas também define um dos objetivos principais que o jogo deseja explorar, que é o trabalho entre os jogadores para alcançar um único objetivo dentro do jogo: defender ou atacar um local específico.

Agora que os objetivos externos do jogo foram definidos, fica mais fácil saber o que precisa ser alcançado e como o *game designer* vai desenvolver as regras, de tal forma que auxilie o jogador a atingir as experiências planejadas no jogo. Mas não são apenas os objetivos externos que são de responsabilidade do *game designer*. Existem, também, os objetivos internos do jogo que são planejados durante o *game design*.

2.2 AS REPONSABILIDADES INTERNAS DO *GAME DESIGN* NO *CS:GO*

Um jogo não é feito apenas de objetivos externos. Ele precisa de regras que façam sentido para o jogador dentro do jogo, pelo menos no mundo virtual onde são implementadas. As regras devem ensinar o jogador quais são os limites da jogabilidade**,** que auxiliam o jogador a alcançar as experiências que os objetivos externos do jogo têm o intuito de atingir. É função do *game designer* desenvolver todas as regras que o jogador vai aceitar se submeter. Em troca, ele recebe experiências recompensadoras planejadas pelo *game design*.

2.2.1 **Escolhendo um lado**

O jogador começa a partida escolhendo um dos dois lados disponíveis: o lado contra terrorista ou o lado terrorista. Depois de ter escolhido um time, o jogador tem à sua disposição um tempo de 60 segundos para fazer um aquecimento e, também, construir uma estratégia com os seus aliados. Esse tempo de pausa, antes de uma partida começar, acontece em todo *round* novo. Dessa forma, os jogadores podem utilizar esse tempo para ir alterando suas estratégias conforme a necessidade, para corrigir um erro ou para melhorar um ataque e uma defesa. É nesse tempo de estratégia que os jogadores têm à disposição uma loja com todos os equipamentos e armas disponíveis para uso.

2.2.2 **O objetivo do jogo**

Para um time alcançar a vitória, alguns objetivos precisam ser atingidos. Eliminar todos os adversários em um *round*, desarmar uma bomba (caso seja do time contra terrorista) ou armar uma bomba (caso seja do time terrorista) são os objetivos que o jogador precisa alcançar para poder vencer. Não é necessário atingir os dois objetivos principais, ou seja, não é preciso eliminar toda a equipe adversária e, ao mesmo tempo, desarmar ou armar uma bomba. Basta conseguir uma dessas tarefas para vencer uma rodada e pontuar no placar.

Uma partida é dividida em trinta *rounds* de um minuto e cinquenta e cinco segundos cada. O time que conseguir vencer quinze rodadas ganha o jogo. Na décima sexta rodada, os jogadores mudam de lado. Quem jogava no time dos contra terroristas, passa a jogar como terrorista e vice-versa. Se cada time ganhar quinze rodadas, isso configura um empate e a partida é encerrada. Esses são os principais objetivos e regras básicas para iniciar uma partida de *Counter-Strike*.

2.2.3 **Sistemas de recompensas e punições**

O sistema de recompensa do *Counter-Strike* visa equilibrar as equipes durante as disputas dos *rounds*. O jogo ajuda o time vencedor e o perdedor ao mesmo tempo. Em cada

rodada, a equipe vencedora recebe um bônus em dinheiro maior que o da equipe perdedora. Dessa forma, tanto o time que vence o *round* quanto o time que perde recebem uma quantia. Quanto maior for a sequência de derrotas consecutivas de uma equipe, maior será o incentivo que ela receberá. Isso ajuda a equilibrar a jogabilidade, incentivando o time derrotado a comprar equipamentos melhores e tentar vencer o próximo *round*.

Existem várias outras maneiras de adquirir recompensas financeiras dentro do *CS:GO*. A arma que o jogador utiliza – para neutralizar um inimigo –, a quantidade de adversários neutralizar pelo jogador – desarmando ou, simplesmente, armando uma bomba – são formas de receber recompensas dentro do jogo.

Cada arma tem uma característica própria, oferecendo vantagens e desvantagens. Fica a critério do jogador escolher o melhor equipamento que atenda a estratégia da sua equipe. Escolher a arma certa para a situação certa faz toda a diferença. Armas que são menos letais ou de pouco alcance são mais difíceis de utilizar para neutralizar um oponente e, por isso, menos eficazes. Elas também deixam o jogador mais vulnerável, obrigando a ter mais cautela durante o *round*. Essas armas precisam oferecer recompensas maiores para o jogador e são essas recompensas que justificam o uso delas.

Mas nem tudo é convertido em bônus para os jogadores. É importante que haja um sistema de punição no jogo, para os jogadores que não conseguem sobreviver até o final do *round*. Por isso, o *Counter-Strike* penaliza jogadores que são eliminados, fazendo eles perderem todo o equipamento adquirido no início da rodada. Os jogadores inimigos e aliados podem coletar itens do chão, deixados por outras pessoas, sejam eles itens largados ou que caíram do inventário de um jogador eliminado. Isso faz com que o jogador tenha medo de perder equipamentos caros, que podem ser coletados pelo inimigo. Consequentemente, essa mecânica ajuda a aumentar a tenção entre as equipes.

2.2.4 **A economia do *Counter-Strike***

Assim como na realidade, o dinheiro, dentro do jogo, é utilizado como ferramenta para comprar e administrar bens. No caso do *Counter-Strike*, serve para adquirir armas e equipamentos de defesa. Em cada *round* vencido, os jogadores são recompensados com um

bônus em dinheiro. Esse bônus é utilizado para a compra de equipamentos de melhor qualidade, tornando o combate mais eficaz.

2.2.5 **Mecânicas de jogabilidade**

Uma das equipes precisa vencer, pelo menos, dezesseis *rounds* para sair vitoriosa numa partida. Existem várias formas de uma equipe vencer um *round* e, com isso, ficar mais perto da vitória final. É possível vencer plantando uma bomba ou desarmando-a; eliminando todos os membros da equipe adversária; ou alcançando esses dois objetivos.

Apesar do *Counter-Strike* ser um jogo em equipe, nada impede de um jogador fazer sua própria estratégia para vencer um *round*. Mas isso tem um custo para sua equipe, que é o enfraquecimento do grupo. As relações sociais em grupo são importantes neste jogo e são refletidas diretamente na jogabilidade.

Quando a partida inicia, no primeiro *round*, todos os jogadores começam com o mesmo valor em dinheiro, com uma arma secundária e com uma arma branca. Fica a critério do jogador adquirir ou não equipamentos de apoio – como coletes ou granadas – para a primeira rodada. O risco de fogo amigo existe e insere um novo nível de jogabilidade. Isso impede que os jogadores atirarem para qualquer lado sem ter algum risco de punição.

2.2.6 **Declarando uma derrota**

Uma forma de terminar o jogo é declarando a derrota. Acontece da seguinte forma: uma votação é aberta para o time e todos da equipe precisam votar “sim” para desistir da partida e, automaticamente, o time adversário é declarado o vencedor. Isso é um recurso disponível para os jogadores quando uma equipe julgar ser impossível virar uma partida.

Essa é a forma como o *Counter-Strike* se caracteriza. Todas essas regras descritas nesse capítulo foram desenvolvidas na etapa do *game design*. E são elas que caracterizam o jogo.

3 **O QUE É *LEVEL DESIGN*?**

3.1 ONDE SE LOCALIZA O *LEVEL DESIGN* NO DESENVOLVIMENTO DE JOGOS?

A produção de um jogo envolve diversas áreas do conhecimento. Para o seu desenvolvimento, é necessário existir, no momento da criação, os produtores, diretores, designers (*game design* e *level design*), programadores, artistas conceituais, escritores, animadores, artistas audiovisuais, profissionais de publicidade e muitas outras áreas. Cada uma delas podendo conter subcategorias e divisões para ajudar na criação de todo material necessário para a criação de um determinado jogo.

A função do produtor é uma das mais importantes entre todas essas áreas listadas. Ele é o responsável por conduzir os profissionais de cada área de atuação e de coletar as informações geradas, pelo seu setor, para depois transmitir as informações para os outros produtores. Cada área pode ter um produtor, que será encarregado de administrar o andamento de suas atividades e objetivos.

Cada jogo tem um orçamento próprio para produção e isso interfere diretamente no tamanho da equipe e, provavelmente, na complexidade do jogo. É completamente possível desenvolver um jogo com uma equipe pequena com dois ou mais desenvolvedores. Mas quando isso acontece a equipe é dividida de uma forma que um profissional acaba exercendo mais de uma função no desenvolvimento. O programador pode ser, também, o escritor ou animador, por exemplo. Essa maneira de criar jogos em equipes pequenas e de forma independente é conhecida como *Indie Games*[3].

Mas onde se localiza o *level design* no meio de todas essas etapas de criação? Os jogos seguem uma ordem natural de criação quando estão em sua fase de desenvolvimento. Tudo começa com uma boa ideia, que pode não ser tão boa assim no final porque ainda precisa ser testada. Essa boa ideia é utilizada como base para desenvolver o Documento de *Game Design* (DGD).

---

[3] Termo em inglês que classifica um jogo feito de forma independente.

Esse documento nada mais é que um arquivo ou livro contendo todas as regras que caracterizam o jogo. Os profissionais, envolvidos no projeto, podem e devem utilizar o DGD como um tipo de guia para tirar dúvidas. Ele serve como um grande livro de consulta. Quando alguém precisa lembrar como funciona um personagem, habilidade ou item, por exemplo, geralmente recorrem a esse documento. Sempre que houver alguma dúvida, relacionada com a jogabilidade ou um visual, o desenvolvedor poderá consultar esse documento.

O trabalho com o *level design* começa logo após a finalização desse Documento de *Game Design*. Só depois de saber quais são as regras definidoras do jogo que é possível começar a pensar e a desenvolver as experiências que o jogador pode ter.

3.2 UMA RELAÇÃO MÚTUA ENTRE *GAME DESIGN* E *LEVEL DESIGN*

Assim como existe uma área do *design* responsável por desenvolver as regras do jogo, também existe uma área responsável por aplicá-las na prática. Essa é a função básica do *level design*, interpretar as regras criadas para o jogo e ensiná-las para o jogador de uma forma que ele se sinta estimulado a continuar explorando tudo aquilo que o jogo tem para oferecer.

Dessa forma, é possível afirmar que *game design* e *level design* tem uma relação mútua de existência; um depende do outro e um acaba complementando a função do outro. Assim, o *game design* representa a teoria e o *level design* representa a aplicação dessa teoria. Essa relação de dependência também é verdadeira quando uma dessas partes é mal desenvolvida. Regras mal elaboradas afetam diretamente no desenvolvimento do *level design.* Basta utilizar regras que não fazem sentido ao mundo onde elas pertencem. Quando aplicadas na prática pelo *level design*, vão parecer que são arbitrárias e sem sentidos. Esse tipo de erro é facilmente percebido pelos jogadores e deve ser sempre evitado.

Depois de termos analisado qual é a função do *game design* e quais são os seus objetivos externos e internos, utilizando o jogo *CS:GO* como base, já temos condições de começar a entender: qual é a relação que o *level design* tem com o *game design*, como essas duas áreas do conhecimento se relacionam e até onde o *game design* influencia no trabalho do *level design*.

Essas duas áreas do desenvolvimento de jogos têm uma relação de codependência, sendo assim:

> Deve estar claro agora que *game design* e *level design* não são a mesma coisa. Também é claro que eles são codependentes e interrelacionados: um é sem serventia sem o outro. A maioria dos jogos não são realizáveis sem algum tipo de *level design*, enquanto que *level design* é uma interpretação das regras do jogo. (KREMERS, 2009, p. 17, tradução nossa)

A relação mútua entre essas duas partes evidencia a necessidade de um trabalho em equipe constante entre *game designer* e *level designer*. Enquanto que o primeiro define as regras que vão caracterizar o jogo, o outro se encarrega de interpretá-las e testá-las afim de conseguir montar um sistema virtual imersivo, cativante, equilibrado, justo e desafiador para o jogador.

Mas a construção desse sistema virtual requer, naturalmente, um trabalho de testes e reformulações das regras. Uma boa ideia no papel não é, necessariamente, uma boa ideia dentro do jogo. Tudo precisa ser testado. Incluir uma regra que possa provocar algum prejuízo na experiência do jogador é algo que precisa ser evitado.

A necessidade de reformular as regras ruins, durante a etapa do *level design*, é mais uma demonstração de codependência com o *game design*. Durante todo o processo de criação do mundo virtual, uma área permanece influenciando na outra.

3.3 AS FUNÇÕES DO *LEVEL DESIGN*

São funções do *level design* criar o ambiente onde o jogo acontece, interpretar as regras do jogo e ensiná-las ao jogador. Uma das principais funções do *level design*, se não a mais importante, é a de ensinar. É dever do *level designer* transmitir, através da jogabilidade, todas as informações que o jogador precisa aprender para conseguir experimentar aquilo que foi projetado em um mundo virtual. Quando jogamos, estamos constantemente aprendendo as regras que definem como o jogo funciona, quais são os padrões existentes nestas regras e qual é o objetivo a ser alcançado e dominado.

É justamente essa característica – a de que os jogos têm de ensinar algo – que torna esse tipo de entretenimento tão atraente para muitas pessoas. Nesse sentido, os:

> Jogos são algo especial e único [...] Uma vez que eles são abstratos e icônicos, eles estão prontos para serem absorvidos. Uma vez que são sistemas formais, eles excluem detalhes extras que distraem. Normalmente, nosso cérebro precisa trabalhar duro para tornar uma realidade confusa em algo tão claro quanto um jogo é.
> Em outras palavras, jogos servem como uma ferramenta fundamental e poderosa de aprendizagem. (KOSTER, 2010, p .36, tradução nossa)

3.4 O CARÁTER EDUCACIONAL DO JOGO

Quando começamos uma aventura em um jogo, tudo é muito novo para a gente. Por isso, é importante que exista uma explicação que justifique tudo o que há dentro do mundo virtual. Se isso não for feito, o jogador perde o interesse facilmente e pode deixar o jogo de lado.

Ele precisa ser conduzido e cativado durante todo o momento da jogabilidade. Precisa aprender como esse mundo virtual funciona, até onde pode ir. Precisa aprender o que pode ser feito e o que não pode ser feito. É necessário ensinar como um item ou habilidade funciona antes de forçar o jogador a utilizar este item ou habilidade, pois não existe algo mais frustrante e desestimulante para alguém do que ser colocado à prova antes mesmo de saber o que precisa ser feito com um item ou habilidade.

Essa forma de conduzir o jogador expõe essa característica educacional dos jogos. Nós estamos sempre aprendendo uma nova mecânica, uma nova habilidade. Isso motiva o jogador a querer aprender cada vez mais algo novo dentro do jogo.

3.5 A IMPORTÂNCIA DA DIVERSÃO

Todo jogo, ou pelo menos a maioria deles, tem como ponto de partida um único objetivo, oferecer um conjunto de experiências positivas e divertidas para os jogadores. É evidente que, cada jogo tem os seus próprios objetivos a serem alcançados e experiências únicas que desejam para o seu público-alvo. Mas todos eles, independente do gênero do jogo, sempre buscam a diversão como resultado a ser alcançado. Ninguém gosta de jogar algo que seja chato, que não cativa o nosso cérebro e que não faça a gente descobrir padrões e resolver problemas.

É exatamente nessa busca em tentar entender padrões e resolver *puzzles* que podemos encontrar a diversão nos jogos. Nada é mais gratificante, para quem está jogando, do que conseguir superar um grande desafio ou entender e solucionar algum problema sugerido pelo jogo. Isso nos mantêm focados na tarefa que estamos fazendo e incentiva o jogador a buscar por mais problemas para serem resolvidos. O nosso cérebro adora solucionar problemas. E quanto mais difíceis eles são, mais desafiado nosso cérebro fica. Consequentemente, a

jogabilidade tende a ficar mais imersiva e isso ajuda ainda mais na aceitação do mundo virtual como um lugar perfeito para testar e aprender coisas novas.

3.6 SISTEMAS DE RECOMPENSAS

Para o *level designer*, o sistema de recompensa deve ser utilizado como um conjunto de ferramentas de engajamento, encorajando ou desencorajando certos comportamentos do jogador, de acordo com os objetivos que o jogo deseja alcançar e, também, com a jogabilidade que o *level design* deseja oferecer para quem está jogando.

3.6.1 **O desejo de escapar da realidade**

A fuga da realidade, ou o escapismo, faz parte da natureza humana. Sempre temos o desejo de explorar um novo mundo ou um mundo de fantasia quando estamos lendo um livro, assistindo um filme ou jogando alguma coisa. Dessa forma, naturalmente, nós nos permitimos escapar para dentro desses mundos. Esse desejo fica ainda maior se for experimentado dentro de um sistema regido por regras pré-definidas que visam garantir a segurança da exploração do ambiente, como os jogos são.

Por este motivo, o *level designer* tem uma vantagem em relação aos profissionais de outras áreas, que é o perfil de seu público. Os jogadores de *videogames* aceitam com mais facilidade um mundo fictício, simulado ou de fantasia. Eles são mais suscetíveis à exploração de um mundo e, eventualmente, aceitam com mais facilidade as regras que regem este lugar a ser explorado. Tudo isso contribui para uma maior imersão que, ao mesmo tempo, facilita no escapismo do jogador para um mundo simulado ou fantástico. Sem o medo de errar. Afinal de contas, mesmo com a pior das punições em um jogo, que é a morte, é só começar e tentar tudo de novo. Quando alguém recorre ao escapismo, essa pessoa está buscando experimentar sensações divertidas de forma segura, sem temer as punições que uma atitude ou decisão pode provocar.

3.6.2 **O jogador gosta de ser desafiado**

A vida está sempre oferecendo desafios para serem superados por nós, que podem se manifestar através de atividades positivas ou negativas. Ao enfrentar um desafio, o resultado pode ser bom e gratificante quando superado. O mais importante nisso tudo é o fortalecimento da moral de quem está enfrentando o desafio.

Nos jogos, os jogadores adoram ser desafiados. Geralmente é o que mais acontece com eles dentro de um mundo virtual e essa é uma excelente forma de cativá-los dentro do jogo. Isso ajuda a exercitar o nosso cérebro, que é faminto por solucionar *puzzles* e que adora ser testado. Por isso, é possível afirmar que:

> Jogos que falham em exercitar o cérebro se tornam chatos. Isso é o que o jogo da velha acaba caindo – É um exercício, mas tão limitado que não precisamos destinar tanto tempo nele. Conforme aprendemos mais padrões, mais novidades são necessárias para deixar o jogo atrativo. (KOSTER, 2010, p. 38, tradução nossa)

3.6.3 **O jogo precisa ser justo**

Uma outra maneira de recompensar o jogador é oferecendo para ele um ambiente regido por regras que são justas. Nada é tão ruim para o jogador do que enfrentar um desafio que ele não está pronto para superar, ou um desafio que é desequilibrado ou impossível de ser vencido. Por isso, é importante ensinar tudo o que o jogo tem para oferecer antes de desafiar o jogador em alguma tarefa.

É igualmente importante testar a jogabilidade, para ter certeza de que não tem uma tarefa mal projetada que, em algum momento, pode impossibilitar a evolução do jogo. O jogador precisa aprender como um item funciona antes mesmo de ser obrigado a utilizar esse item em algum conflito. Precisa aprender até onde ele pode explorar um cenário, o que é passivo de interação e o que não é, se pode pular de certa altura sem levar dano ou se já tem o nível e itens necessários para enfrentar um determinado inimigo.

Todas essas informações precisam ser apresentas para o jogador antes de o jogo oferecer qualquer tipo de desafio. Ele precisa estar preparado, se não, um sentimento de injustiça pode ser desenvolvido pelo jogador.

Quando não ocorre a superação do desafio, não conseguimos atingir aquilo que nos motiva a continuar com a jogabilidade. Quando não somos recompensados com a alegria da

vitória ou com o sentimento de superação de uma tarefa, temos a tendência de perder o interesse no jogo. Ou então, podemos passar a encarar o desafio oferecido como algo negativo que devemos evitar a qualquer custo.

Tudo fica ainda pior se o motivo de não conseguir a superação desse desafio for provocado por uma tarefa mal projetada e, consequentemente, desequilibrada. O sentimento de injustiça é quase que garantido e é muito importante evitar esse tipo de sensação nos jogadores.

3.6.4 **Uma recompensa é sempre bem-vinda**

Todo esforço do jogador merece e deve ser recompensado. Talvez essa seja a maior ou, pelo menos, uma das mais importantes motivações que justifica o interesse e a permanência do jogador em uma determinada tarefa. Quando alguém se submete a um desafio, seja ele fácil ou muito difícil, é sempre esperado que o jogo ofereça algo em troca. Na maioria das vezes, não importa se a recompensa é um item colecionável ou um item super raro. O que realmente importa, para o jogador, é receber algo em troca que esteja à altura do seu esforço. E é importante esse ponto: porque recompensar um esforço oferecendo qualquer coisa em troca tem o mesmo valor que não receber nada.

Uma ótima maneira de explorar esse sistema de recompensa é dando para o jogador alternativas de exploração do cenário. Oferecer diversos pontos de interesse para serem visitados e descobertos, durante a exploração do mundo virtual, é uma maneira muito eficaz de desenvolver o espaço de jogabilidade do cenário. E recompensar essa exploração, feita por ele, com itens interessantes ou qualquer outro tipo de colecionável, que só poderia ser achado caso o jogador fizesse esses caminhos alternativos, são ótimas formas de fazer com que o mundo virtual ganhe ainda mais riqueza e importância, facilitando ainda mais na imersão do jogador.

3.7 ESPAÇO DE JOGABILIDADE

Já é possível perceber que faz parte da função do *level design* desenvolver, não apenas a jogabilidade que o jogo deseja oferecer, mas, também, o espaço virtual em que ela será vivenciada pelos jogadores. Quando o *level design* está em desenvolvimento, uma de suas principais preocupações é a criação de um ambiente virtual com um bom espaço para

exploração e uma boa jogabilidade. É onde tudo acontece, onde se localiza tudo aquilo que poderá ser utilizado, explorado e testado pelo jogador durante o jogo.

O desenvolvimento desse espaço é definido pelas regras criadas no *game design* e pelos testes e ajustes feitos no *level design*, afim de oferecer a melhor jogabilidade possível em um ambiente seguro para o jogador, onde poderá testar estratégias e aprender a superar desafios. Ao atuarmos:

> [...] como *level designers* podemos moldar o mundo e o cenário de jogabilidade contidos nele. Podemos oferecer uma mão manipulando o jogo para o jogador, eliminando muitas das injustiças inerente à vida real, e oferecer aos jogadores múltiplas chances de superar o mesmo desafio. (KREMERS, 2009, p. 117, tradução nossa)

3.8 IMERSÃO E SUSPENSÃO DE DESCRENÇA

Sempre que aceitamos jogar algum jogo, seja qual for o gênero dele, uma simulação hiper-realista ou um jogo de fantasia, nós aceitamos acreditar em mundos virtuais que não existem de verdade. Aceitamos porque, ao atingir uma suspensão de descrença em relação a um mundo fictício, podemos acreditar nas regras que definem esse mundo com mais facilidade e assim aceitando elas como verdades absolutas dentro do mundo do jogo. Dessa forma, podemos experimentar algo que não seria possível no nosso dia a dia, por conta do mundo real punir severamente as pessoas quando cometem algum erro ou arriscam suas vidas em desafios.

Para poder escapar da realidade, antes temos que aceitar ou fazer de conta que o mundo virtual, que usamos para jogar, existe mesmo que momentaneamente durante algumas horas de jogabilidade. Isso só conseguimos alcançar tendo uma suspenção de descrença em relação a tudo que define o mundo virtual do jogo para, só assim, conseguirmos atingir uma certa imersão no jogo explorando o que o jogador pode fazer de divertido, sempre focando em atingir experiência positivas.

Nas artes, nos filmes, nos livros e, consequentemente também, nos jogos a imersão e a suspensão de descrença sempre foram utilizados como formas de manipular uma audiência e um público-alvo. É do interesse do *level designer* conseguir conduzir o jogador durante todo o momento da jogabilidade. Só assim poderá ter certeza de que conseguiu atingir tudo aquilo que o jogo tem para oferecer.

Por causa dessa característica manipuladora que o *level design* pode ter sobre as experiências do seu público – um público que já aceita com facilidade ser conduzido –, é importante ter em mente que podemos influenciar negativamente ou positivamente tudo aquilo que é vivenciado pelo jogador. Isso torna a tarefa de criar experiências de jogabilidade ainda mais difícil. Porém, nunca é demais lembrar que "não há nada de sinistro em um contador de histórias habilidoso cativar uma audiência em torno de uma fogueira. Nós não nos ressentimos com um cineasta por utilizar ângulos fora do comum para conseguir expressar a força de uma cena." (KREMERS, 2009, p. 140, tradução nossa)

O objetivo da imersão é conseguir auxiliar o jogador a alcançar as experiências positivas que existem nos jogos, com a ajuda da suspenção de descrença, que facilita a aceitação de um mundo que não existe. Só assim esse mundo fictício pode ser considerado real durante a jogabilidade. A experiência com o jogo, portanto, deve promover nos jogadores a crença "[...] nesses mundos como consequência das regras formais do jogo. Se o *level design* atua como um mecanismo de ensino, isso ajuda se o aluno estiver disposto a acreditar no professor." (KREMERS, 2009, p. 143, tradução nossa)

4 *LEVEL DESIGN* APLICADO NO *CS:GO*

4.1 PESQUISA DE REFERÊNCIA PARA O CENÁRIO

Toda parte prática do projeto teve como referência pontos da Cidade do Rio de Janeiro. E, para isso ser possível, foi utilizada a ferramenta *Google Street View* – um excelente recurso para caminhar virtualmente pela cidade e conseguir capturar um pouco de sua arquitetura.

Abaixo colocamos um conjunto de imagens que serviram de apoio para a montagem do cenário. São pedaços do Rio de Janeiro com características únicas e que despertaram minha curiosidade em testá-los dentro de um ambiente virtual de um jogo.

Figura 1 - Referências para o cenário

Fonte: Google Street View

Figura 2 - Cenário 3D inspirado na cidade do Rio de Janeiro

4.2 DESENVOLVIMENTO DOS *ASSETS*

Uma das melhores formas de iniciar qualquer criação, seja de itens do jogo ou artes de apoio, é indo atrás de boas referências visuais, que ajudam a auxiliar o desenvolvimento do objeto em questão.

Quando nosso objetivo é desenvolver um *asset*[4] 3D, é necessário, além de boas referências de apoio, um conjunto de atividades. Essas atividades, que são quase todas elas obrigatórias para a criação do objeto em desenvolvimento, formam e preparam o *asset* para ser inserido dentro de algum motor gráfico[5]. Seja qual for o motor gráfico utilizado pelo desenvolvedor, ele terá que respeitar algumas regras de criação de objeto 3D. Porque, não basta

[4] É um objeto utilizado no desenvolvimento de um jogo. Ele pode ser um áudio, uma imagem, um objeto 3D ou até mesmo um texto.
[5] Ferramenta/ambiente onde se desenvolve um determinado jogo.

modelar o *asset* em um programa de modelagem, o *designer* precisa tomar alguns cuidados e ter sua atenção em certas partes do modelo 3D que está desenvolvendo.

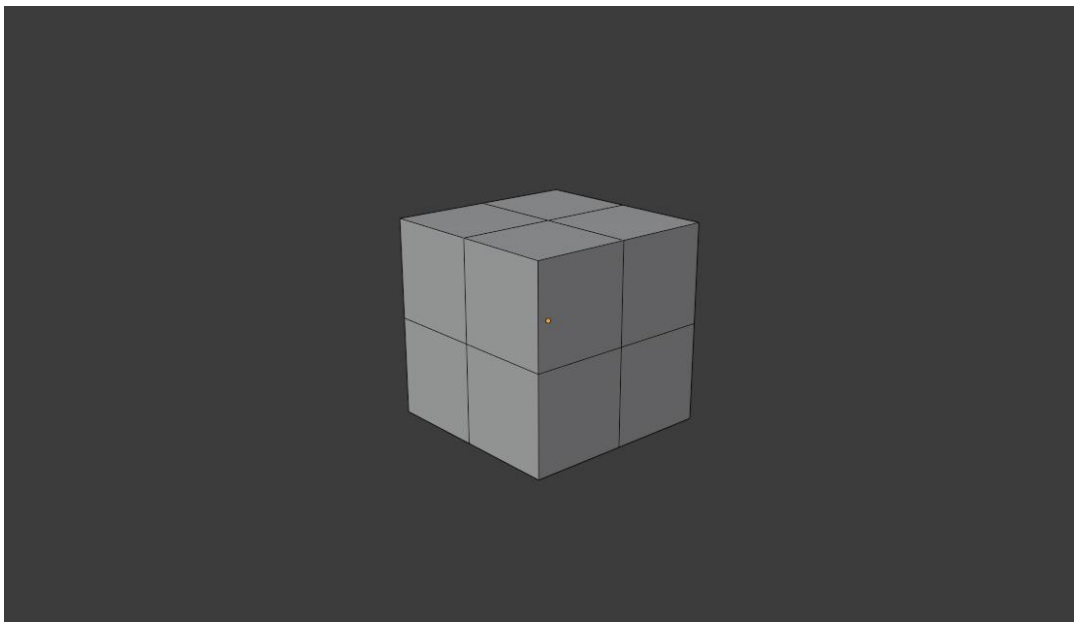
As características que todo desenvolvedor de *asset* 3D precisa ter atenção são: topologia do objeto, *UV Unwrap*, *Texel Density*, orientação da face de um objeto e a proporção do objeto.

4.2.1 **Topologia**

Quando falamos em topologia estamos nos referindo à malha 3D que compõe o objeto tridimensional. Alguns cuidados importantes merecem destaque quanto à topologia.

No mundo da computação gráfica o ideal para a malha 3D de um objeto é ela ser composta por apenas faces quadriculares. Como pode ser visto, abaixo, na imagem.

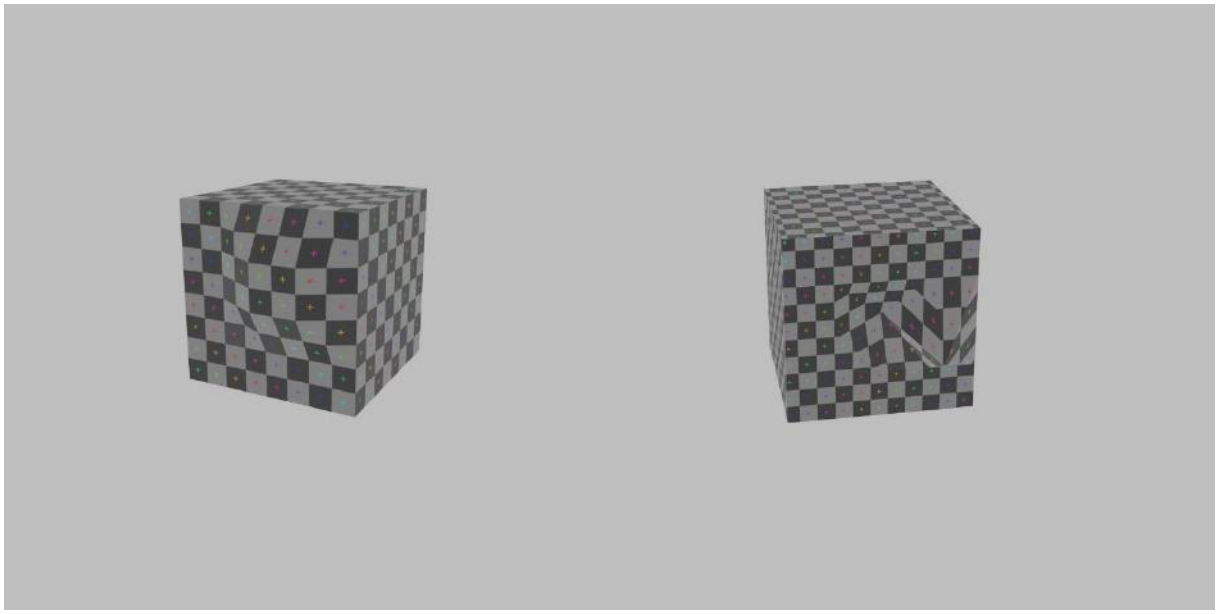
Figura 3 - Malha 3D com faces quadriculadas



Esse ideal de malha 3d é muito importante para a criação de qualquer *asset*, já que, quando nos deparamos com faces triangulares ou faces com mais de 4 vértices, é quase certeza de que o objeto 3D terá algum tipo de problema na texturização ou na animação, quando sofrer algum tipo de deformação em sua malha 3D.

A seguir, temos exemplos de um objeto com faces quadriculares (à esquerda) ao lado de um outro com faces triangulares (à direita), ambos com deformação na malha.

Figura 4 - Malhas 3D com deformação



É possível notar, mesmo que de forma sutil, uma diferença na deformação entre faces de objetos quadriculares e faces de objetos triangulares. É muito mais natural e com um resultado muito mais próximo do desejável, haver deformações em uma topologia composta por apenas faces com 4 vértices.

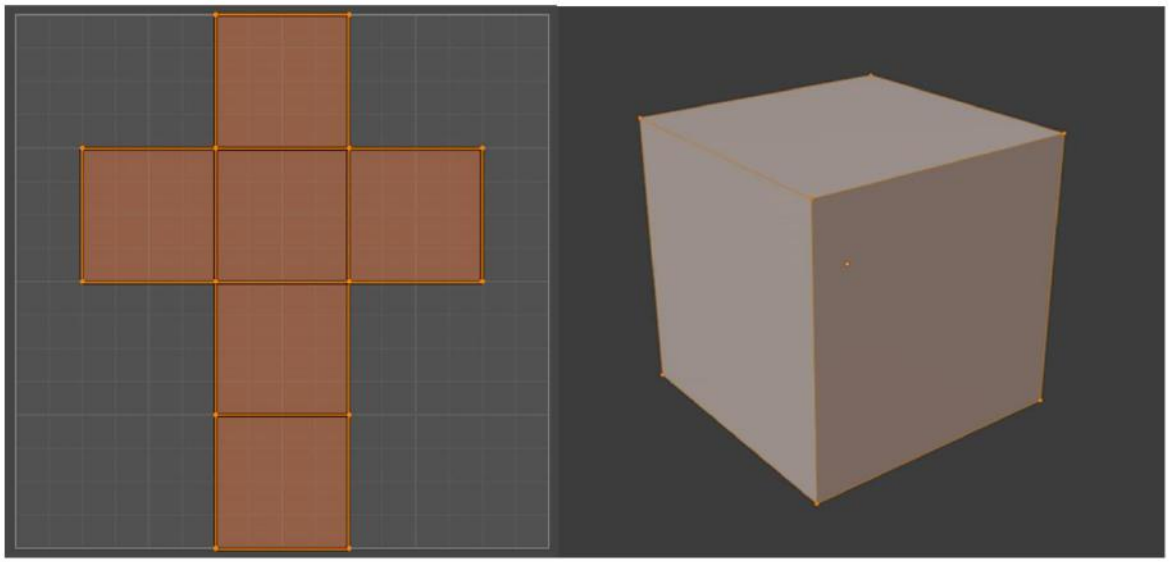
Um outro ponto muito importante sobre topologia de objetos é a sua quantidade de faces e vértices. É muito importante, para um desenvolvedor, conseguir desenvolver um objeto 3D da melhor qualidade possível usando a menor quantidade de polígonos que conseguir.

Isso se dá por conta da limitação que motores gráficos têm em processar uma grande quantidade de polígonos em tempo real. É muito custoso, para qualquer tipo de *hardware*, conseguir processar milhões de vértices e faces. Por este motivo, a quantidade de polígono que um *asset* tem afetará diretamente no desempenho de um jogo e, consequentemente, na experiência do jogador.

Depois de tentar construir um *asset* com uma boa malha – na melhor qualidade visual possível e com uma quantidade de polígonos aceitável –, o próximo passo é iniciar a preparação da malha 3D para a texturização.

UV Unwrapping é uma técnica em computação gráfica que se encarrega de representar uma textura bidimensional em uma malha tridimensional. O processo pode ser demorado e difícil, em alguns momentos, mas é obrigatório para qualquer objeto 3D que vai receber algum tipo de texturização. A imagem abaixo exemplifica o que basicamente é feito neste processo de UV *UV Unwrapping*.

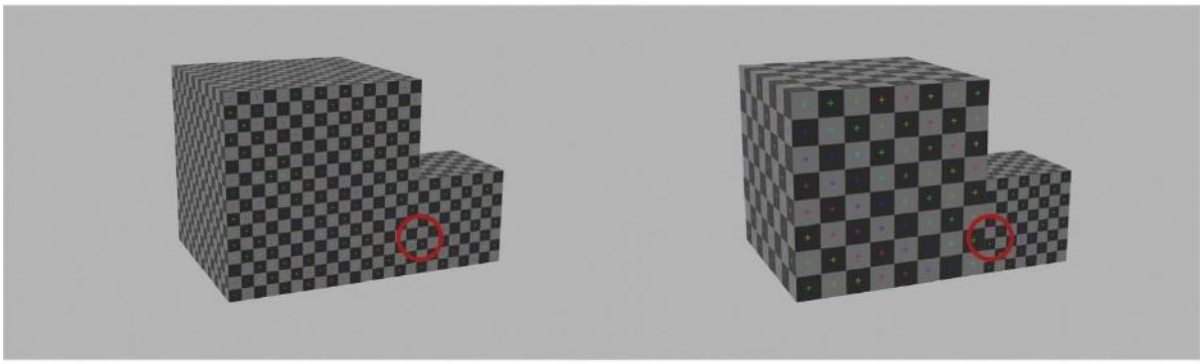
Figura 5 - *UV Unwrapping*



Além do *UV Unwrapping*, temos o *Texel Density* que, também, tem relação direta com a texturização de *assets* e afeta diretamente no visual dos objetos nos jogos. Apesar de ter influência dentro de um jogo, o *Texel Density* pode passar despercebido, na maioria das vezes. O ideal é que aconteça exatamente isso, mas, quando ele é notado – muita das vezes pelo próprio jogador –, provoca uma quebra na imersão, mesmo que por um breve momento.

Abaixo temos dois exemplos. Um (à esquerda) é considerado como uma boa aplicação do *Texel Density* e o outro (à direta) é considerado um como algo a ser evitado.

Figura 6 - Densidade de *pixel* boa X ruim

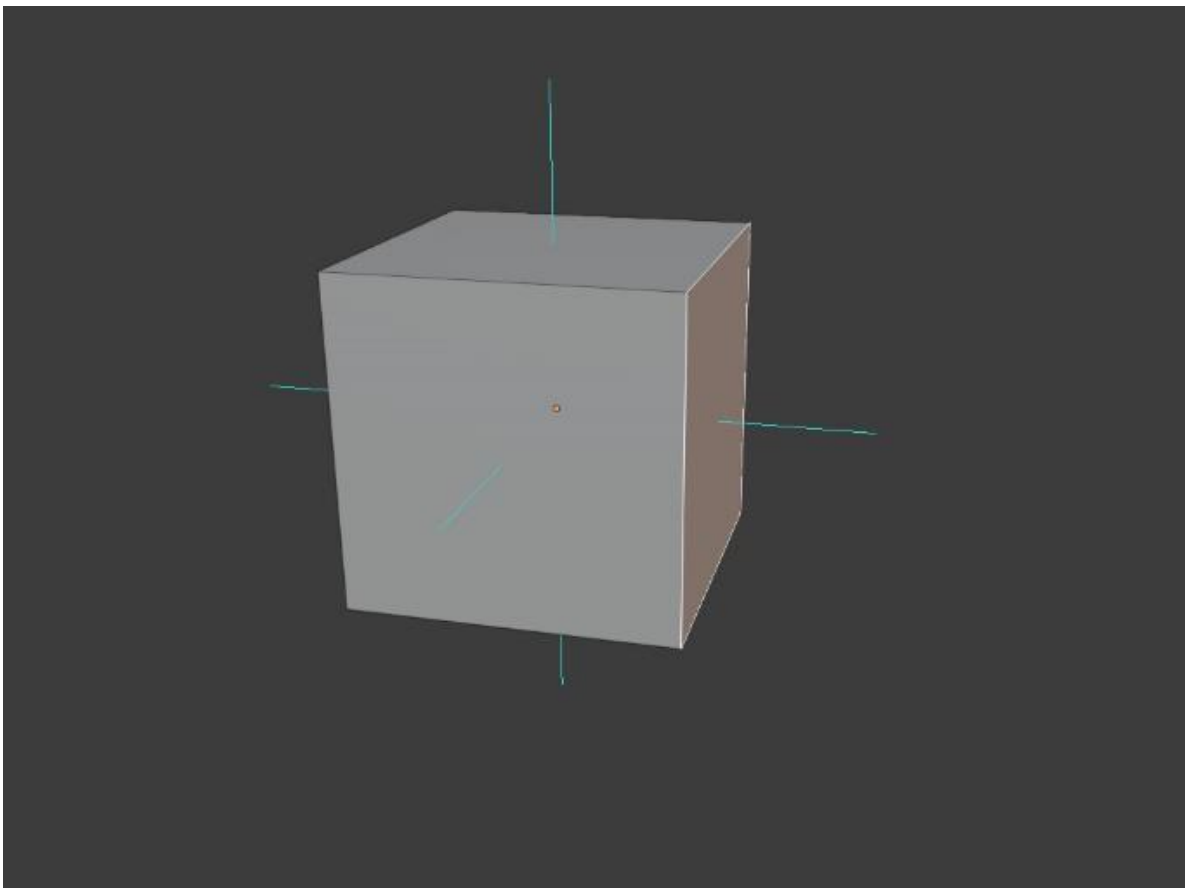


Note que, na figura à direita, a área marcada em vermelho demonstra uma grande diferença de resolução das texturas entre os dois objetos.

Na maioria dos casos, esse tipo de problema passa sem ser notado por ninguém, até mesmo pelo próprio desenvolvedor. Quando é o jogador que percebe a falha, o reflexo pode recair sobre sua imersão do jogo. E é essa a razão pela qual a etapa de verificação do *Texel Density* dos *assets* é tão importante. Evitar a quebra da imersão dos jogadores a todo custo é fundamental.

Mais um ponto importante, que não pode ser deixado de lado – e que influência diretamente na texturização de um objeto –, é a orientação das faces do *asset*. Os chamados *normals* são uma indicação da orientação de uma face. É para onde a face de um objeto está virada. Se os *normals* estiverem invertidos, a face vai parecer que não existe. Todo programa 3D interpreta *normals* invertidos como faces invisíveis. A seguir, temos uma representação dos normals das faces de um objeto, indicados por uma linha azul:

Figura 7 - Orientação das faces de um *mesh*



Por último, por se tratar de um ponto mais fácil de ser resolvido, podemos destacar a necessidade de criar *assets* numa proporção próxima à dos objetos da vida real. Um dos motivos para ter esse cuidado é o fato dos motores gráficos dos jogos usarem física reais para simular um mundo virtual – mesmo sendo esse mundo um mundo de fantasia. Fica mais fácil para o motor gráfico calcular a física das coisas quando tudo respeita um certo nível de proporção. Assim, mesmo sendo um cuidado fácil de ser resolvido dentro de qualquer programa 3D, é necessário ter atenção quanto à proporção dos *assets* no momento do desenvolvimento.

4.2.2 **Texturização**

A texturização é o processo de utilizar imagens bidimensionais em objetos tridimensionais a partir do mapeamento da malha 3D feita na etapa de *UV Unwrapping.* Esse processo de criação não se limita apenas a inserir uma imagem em um objeto. Existem algumas

etapas de elaboração da texturização que resulta em mapas como o *base color*, *normal*, *height*, *roughness* e *metalic*. Sendo, cada um desses mapas, responsável por representar características fisicamente fiéis de um determinado material. Dessa forma, a texturização não é apenas composta por uma imagem representando única e exclusivamente o canal de cor.

A textura é representada por n*ormal* e *height* que influenciam na superfície alterando o relevo, criando cavidades e, também, sobressalências na superfície do objeto. Além desses dois mapas, temos também outros que influenciam em como a superfície de um objeto reflete a luz – como é no caso do *roughness*, que atua alterando o quão reflexivo uma superfície pode ser. Já o mapa *metalic*, se encarrega de informar o que é metal e o que não é metal no material.

Juntos esses mapas auxiliam na representação de um material fisicamente preciso e com muito mais controle no momento de desenvolvimento da textura. Assim, é mais fácil manter a textura fiel em qualquer ponto do cenário 3D. Hoje a textura é uma junção de todos esses mapas com cada um deles sendo responsável por simular uma característica específica, seja qual for o material simulado.

Abaixo podemos ver a representação do *base color*, *roughness*, *metalic*, *height e normal*, respectivamente.

Figura 8 - *Base color*

Figura 9 - *Roughness*

Figura 10 - *Metalic*

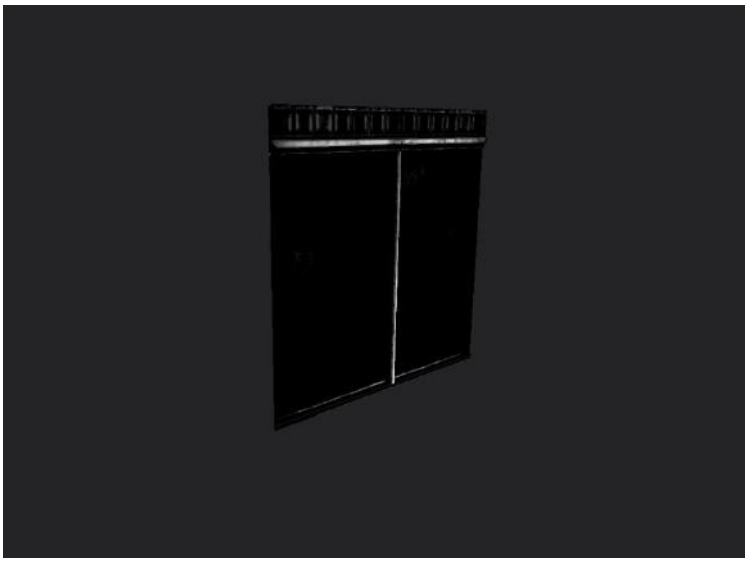


Figura 11 - *Height*

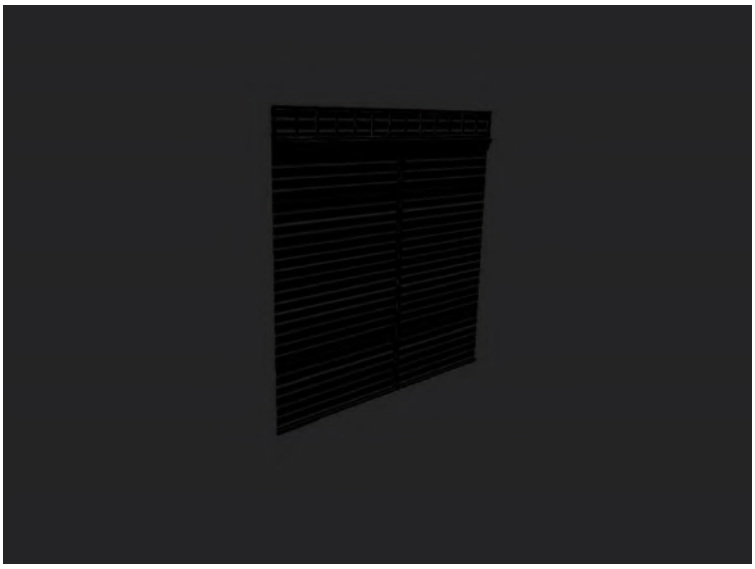
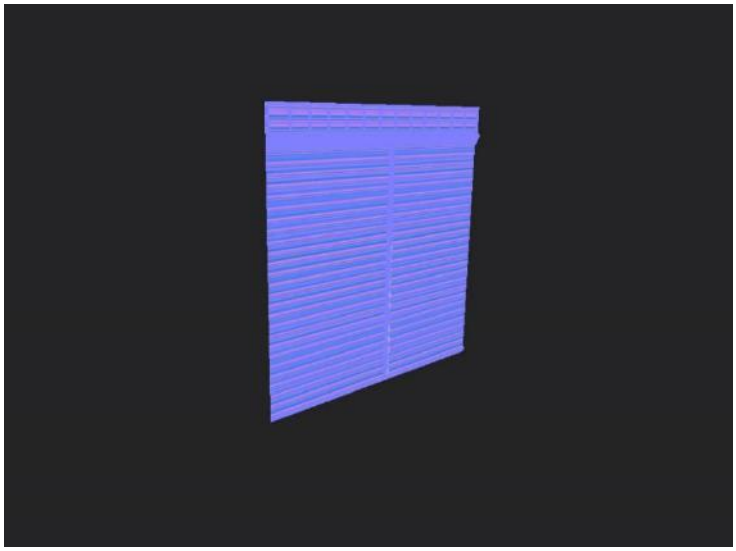


Figura 12 - *Normal*



Apesar dos mapas *roughness, metalic e height* serem em preto e branco eles exercem funções completamente diferentes. O resultado a seguir ilustra o poder desses mapas quando trabalhados juntos para gerar um único resultado que é conhecido no desenvolvimento dos jogos como material.

Figura 13 - Objeto resultado da composição dos mapas

4.3 PROTOTIPAGEM E *BLOCKING*

A prototipagem é uma etapa de grande utilidade para o *level designer*. É nesse momento do desenvolvimento que as regras do jogo são testadas. O jogo precisa passar por análises até o *level designer* ter certeza de que está na direção certa para conseguir oferecer a melhor jogabilidade e imersão para os jogadores.

Apesar de não ser um momento muito rico, em desenvolvimento estético do cenário, essa é uma etapa que consegue gerar muitas soluções de design que no papel não seria possível prever e nem solucionar. A etapa de teste das regras é rica em desenvolvimento do equilíbrio do jogo e entendimento de como o jogador pode se divertir no espaço de jogabilidade criado.

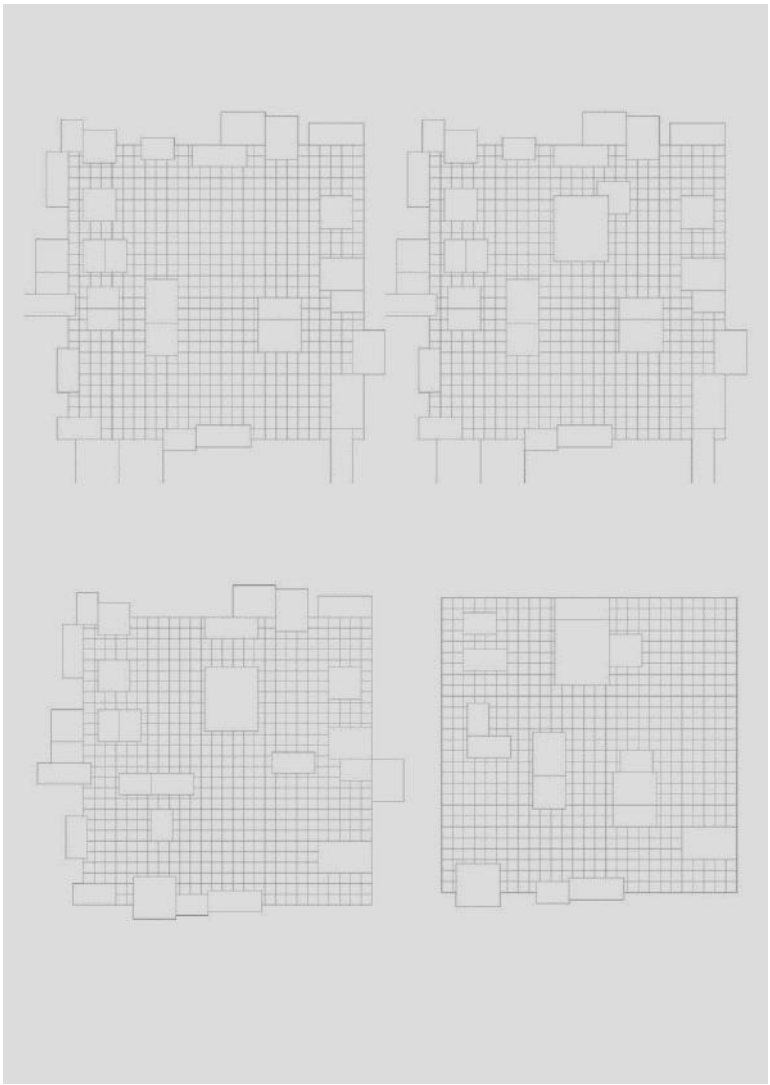
O *blocking* nada mais é do que uma forma simplificada de criar um ambiente jogável minimamente próximo do cenário que será desenvolvido, ou seja, é uma versão simplificada do cenário que, posteriormente será mais bem desenvolvido na etapa de composição e desenvolvimento da estética. Nesse momento, a única preocupação do *level designer* tem que ser com a jogabilidade e desenvolvimento do espaço de jogabilidade.

Uma boa forma de iniciar esses testes é esboçar algumas ideias no papel, em forma de blocos, e posteriormente aplicar esses esboços na prática; ou seja, criar o cenário em 3D e testar o quanto de espaço o jogador terá para interagir – o com o próprio cenário e com outros

jogadores – e verificar a distribuição de elementos que não terão nenhum tipo de interação, mas que vão servir como cobertura ou bloqueio para alguma ação do jogador.

As imagens abaixo são estudos iniciais, feitos no papel, com a tentativa de elaborar um início de desenvolvimento de cenário. Apenas se preocupando com a distribuição dos elementos que o ocupam mais espaço no ambiente.

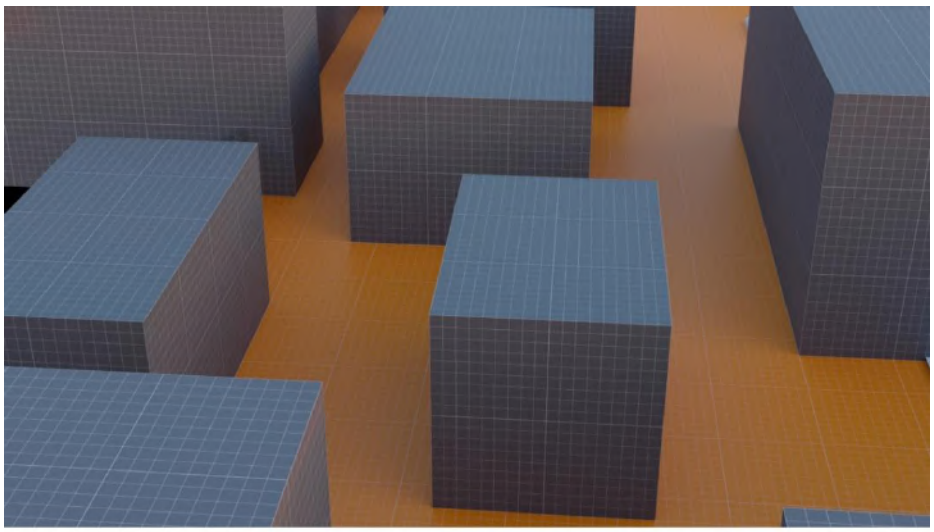
Figura 14 - Esboço de mapa

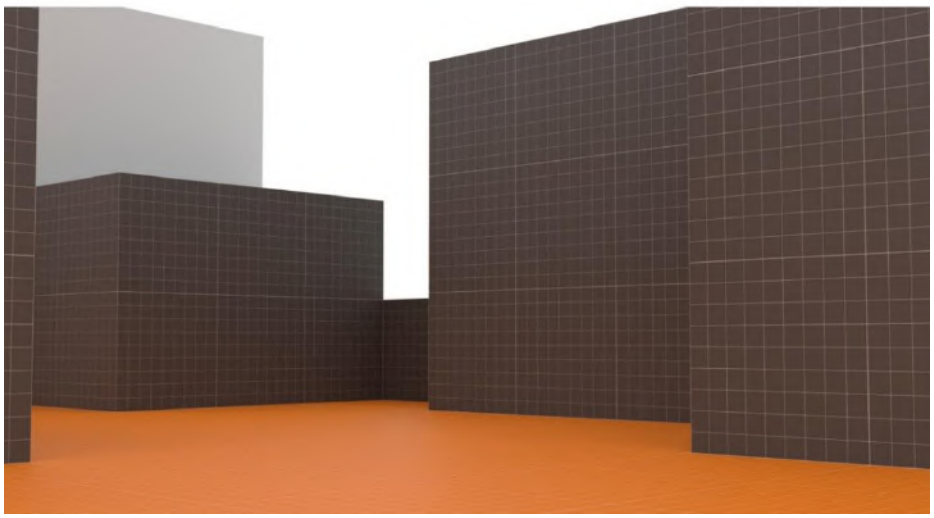


Alguns testes com protótipos sempre ajudam a enxergar detalhes que, no papel, pode passar despercebido. Aqui um exemplo que ajudou a entender como seria distribuído os blocos de edifícios, qual seria a altura máxima desses edifícios, quantos caminhos cada equipe teria a

disposição para chegar no mesmo local, onde cada equipe iniciaria no mapa e quais seriam os possíveis pontos de conflito no cenário.

Figura 15 - Protótipo de cenário



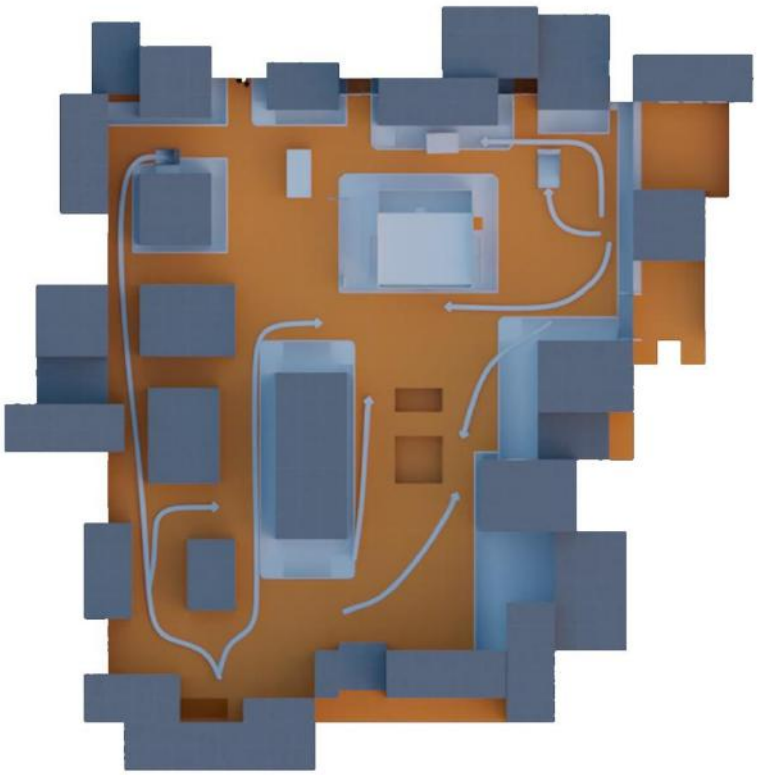


Na imagem acima, ficou claro que os prédios – que estão no meio do espaço onde os jogadores teriam interação com o próprio cenário e outros jogadores – não poderiam ser muito

alto a ponto de impedir lançamentos de granadas. Trata-se de uma mecânica muito utilizada no CS:GO e que não poderia ser prejudicada com um escolha errada de *level design* dos edifícios.

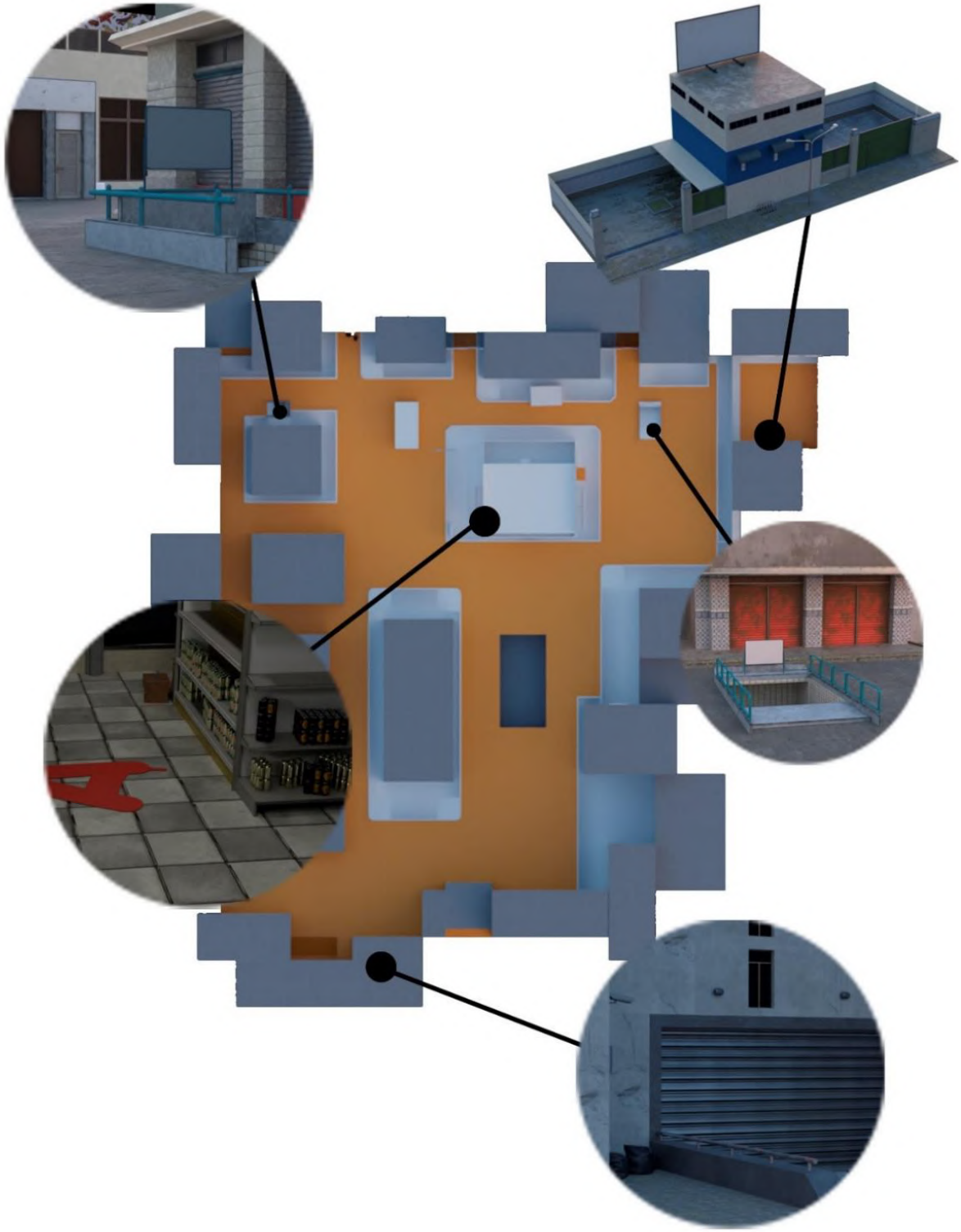
Abaixo temos uma demonstração do espaço disponível para o jogador explorar e interagir com outros jogadores. Quais caminhos estão disponíveis para chegar a um determinado ponto de interesse.

Figura 16 - Possíveis rotas dos jogadores



Por fim, foi possível determinar onde cada equipe iria iniciar em cada rodada, onde seria disposto o local que a equipe contra-terrorista teria que defender e a equipe terrorista deveria atacar e onde seriam localizadas as entradas do metrô.

Figura 17 - Pontos de interesse do cenário



4.4 COMPOSIÇÃO DO CENÁRIO

Após definir o *level* do cenário e o espeço de jogabilidade, temos a tarefa de desenvolver a composição visual do mapa e sua estética. É função do *level designer* desenvolver não só as experiências de jogabilidade, mas também as experiências visuais do jogador.

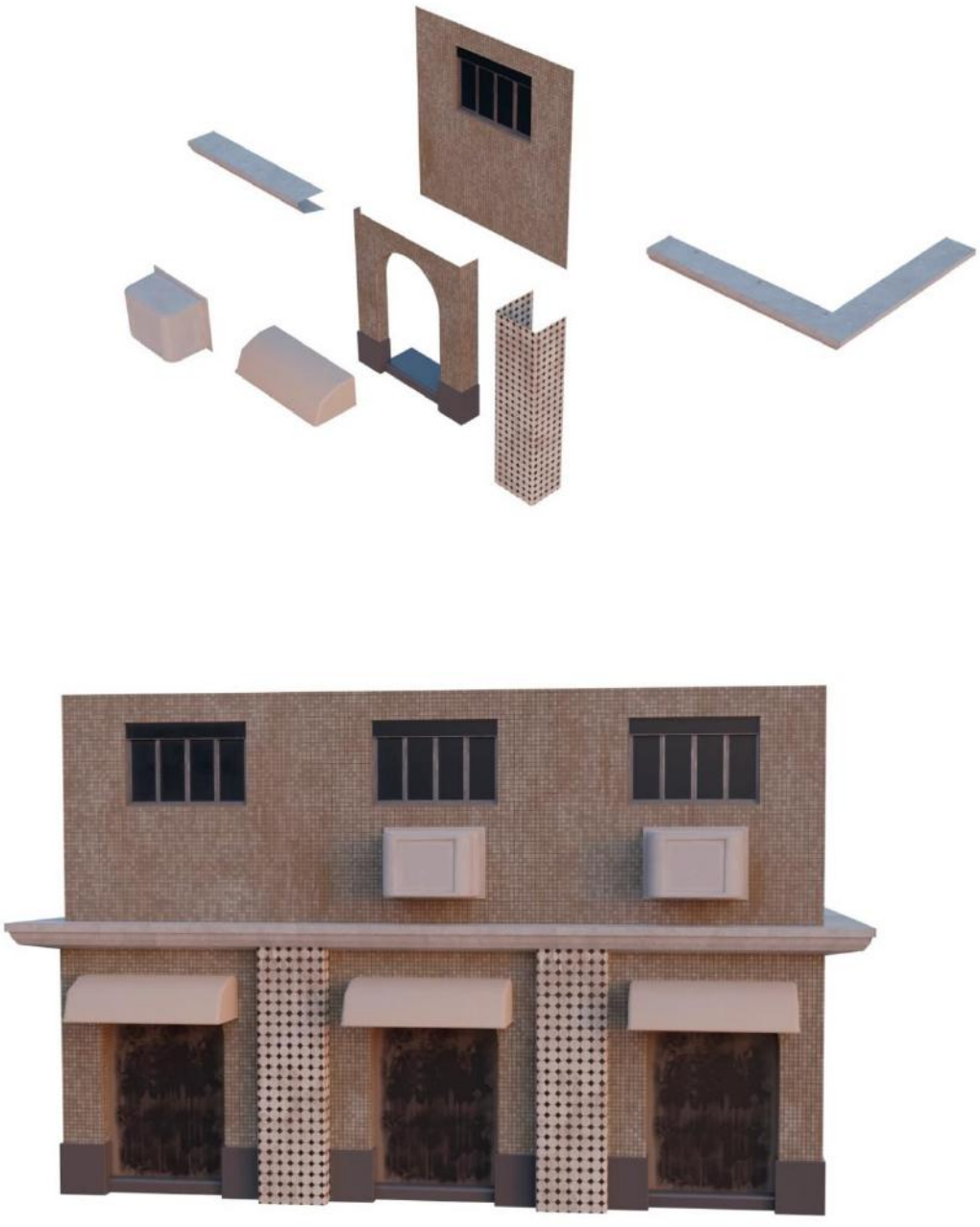
Apesar da etapa de composição e elaboração da estética do cenário serem, normalmente, feitos depois de definir bem como jogador vai jogar e o que ele poderá experimentar, nada impede dos *assets* serem desenvolvidos em paralelo com a prototipagem do jogo.

4.4.1 **Desenvolvimento de *assets* modulares e a reutilização de objetos**

Umas das técnicas mais utilizadas na criação de cenários, no mundo dos jogos, é a criação de objetos modulares. Eles são objetos com um grande nível de reutilização, pela simples redistribuição desses *assets* – explorando formas diferentes de conectar um pequeno grupo de objetos, afim de criar outros completamente novos. Esses *assets* modulares funcionam de forma similar com o brinquedo Lego: há um grupo limitado de peças, mas com um potencial de uma peça se conectar na outra, permitindo criar outros objetos inteiramente novos.

Abaixo há uns exemplos do potencial de produção que um *asset* modular pode oferecer para a agilidade do processo de desenvolvimento de um jogo.

Figura 18 - Aplicação de *assets* modular



Em um desenvolvimento de um jogo, é muito comum utilizar esses tipos de artifícios para aliviar a carga de produção em cima dos desenvolvedores. Mas é um recurso que precisa ser usado com cautela, para não sobrecarregar o ambiente do cenário com muitos itens parecidos. É importante haver variações não só no modelo 3D, mas também nas texturas e, inclusive, na forma como esse *asset* vai aparecer no cenário. Uma simples alteração na forma como *asset* é disposto na cena já ajuda a quebrar uma possível homogeneidade que esses itens repetidos podem ter.

A foto abaixo ilustra bem essa questão:

Figura 19 - Reutilização de *assets*

5 **CONCLUSÃO**

A complexidade do desenvolvimento de jogos é refletida na existência das diversas áreas atuantes, da concepção de um projeto até a finalização de um jogo. Este trabalho, ao focar no *level design*, tem a tarefa de criar o desenvolvimento da jogabilidade que o jogador pode ter no meu cenário. Outra questão importante é que seria uma tarefa árdua demais atuar em mais de um setor, em tão pouco tempo disponível, para a finalização do atual projeto. Por este motivo foi determinado que seria abordado *level design* como tema. Usar o Counter-Strike, como base para o meu *level design,* foi uma decisão acertada por se tratar de um jogo conhecido e que oferece uma excelente base pronta de *game design*.

Essa escolha de caminho garantiu, então, que o trabalho com o *level design* não sofresse influências negativas da etapa anterior do processo de desenvolvimento; que poderia ter surgido, por exemplo, com um jogo recém-criado, cujo número insuficiente de testagem pudesse comprometer o equilíbrio do jogo – por haver, por exemplo, uma regra muito fácil que pudesse tirar o interesse do jogador.

Outra escolha acertada no projeto foi determinar um tema para o cenário. Quando trabalhamos na criação de alguma coisa – seja um projeto grande ou pequeno –, ir em busca de um tema, para servir como base para o desenvolvimento do *level design,* é sempre um bom começo. Isso ajuda na pesquisa de referências e a dar um impulso inicial no projeto. Acredito que ter a cidade do Rio de Janeiro como cenário ajuda a despertar o interesse de jogadores, pois, é um tema pouco explorado nos jogos em geral.

REFERÊNCIAS

KOSTER, Raph. **A theory of fun for game design**. Arizona: Paraglyph Press, 2005.

KREMERS, Rudolf. **Level design**: concept, theory and practice. Flórida: CRC Press, 2009.

RABIN, Steve. (editor). **Introdução ao desenvolvimento de games**: entendendo o universo dos jogos. São Paulo: Cengage Learning, 2011.

SCHELL, Jesse. **The art of game design**: a book of lenses. 2. ed. Flórida: CRC Press, 2015.

GLOSSÁRIO

Asset: É um objeto utilizado no desenvolvimento de um jogo. Ele pode ser um áudio, uma imagem, um objeto 3D ou até mesmo um texto.

CS: é um acrônimo da palavra em inglês Counter-Strike.

CS:GO: é um acrônimo da palavra em inglês Counter-Strike: Global Offensive.

E-sports: termo utilizado para definir modalidade esportiva digital.

Indie Games: termo em inglês que classifica um jogo feito de forma independente.

Mod: é uma abreviatura da palavra em inglês *modification*. Os Mods são modificações, feitas geralmente por fãs, que alteram o conteúdo do jogo para gerar novas experiências de jogabilidade.

Motor gráfico: ferramenta/ambiente onde se desenvolve um determinado jogo.