Procedural Content Generation using Artificial Intelligence for Unique Virtual Reality Game Experiences
Geração Procedural de Conteúdo utilizando Inteligência Artificial para Experiências Únicas de Jogo com Realidade Virtual

João Pedro Assunção Campos
Universidade de Passo Fundo
Passo Fundo, RS, Brazil
jp_ascampos@hotmail.com

Rafael Rieder
Universidade de Passo Fundo
Passo Fundo, RS, Brazil
rieder@upf.br

***Abstract*—The aim of this study is to define and develop a procedural content generation method based on Artificial Intelligence for immersive games. We investigated the use of procedural content generation techniques based on agents and decision trees, applying it to a virtual reality game design using Unreal. In this context, it is possible to offer a unique game experience for each user, considering personal decisions during the interaction process.**

***Index Terms*—Procedural Content Generation, Virtual Reality Game, Decision Trees.**

***Resumo*—O objetivo deste estudo é definir e implementar um método de geração de conteúdo procedural baseada em Inteligência Artificial para games imersivos. Para tanto, investigou-se o uso de técnicas de geração de conteúdo procedural baseada em agentes e árvores de decisão, aplicadas para um projeto de jogo com suporte à Realidade Virtual utilizando a Unreal Engine. Desta forma, pode-se ofertar uma experiência diferenciada de aplicação, considerando escolhas do jogador durante o processo de jogo.**

***Palavras-chave*—Árvores de Decisão, Geração de Conteúdo Procedural, Jogo com Realidade Virtual.**

I. Introdução

Geração de conteúdo procedural (*Procedural Content Generation*, PCG) se refere à criação algorítmica de conteúdo utilizando um procedimento aleatório ou pseudo-aleatório que resulta em uma variedade imprevisível de elementos [1]. Em jogos, por exemplo, pode-se criar durante o processo interativo estruturas como mobílias, pinturas, estátuas e mapas, além de elementos de arquitetura, história e poesia, música e dança, conversação, comportamentos, decisões ou estratégias. A PCG permite que tais conteúdos possam ser gerados automaticamente, reduzindo a carga de trabalho de artistas e desenvolvedores, assim como espaço em disco para os jogos.

Métodos de geração de conteúdo procedural vêm gradualmente se tornando prática comum na indústria de jogos. Eles podem ser vistos em títulos famosos como Minecraft (https://www.minecraft.net) e No Man's Sky (https://www.nomanssky.com), onde universo de jogo e mundos exploráveis pelo jogador são gerados a partir de algoritmos de criação de mundos dinâmicos, possibilitando uma infinidade de possibilidades exploratórias ao jogador [2].

Todavia, a PCG é utilizada em contextos muito restritos e específicos dentro dos elementos que compõem os jogos. O uso não tão amplo das técnicas de PCG pode ter relação com a sua imprevisibilidade, pois designers não desejam que a aplicação gere erroneamente um conteúdo essencial do jogo. Logo, a PCG é raramente implementada para gerar níveis inteiros de jogos, como visto na criação de masmorras, um tipo específico de nível de jogo muito encontrado em títulos de aventuras e RPGs [3].

O âmbito de Inteligência Artificial (*Artificial Intelligence*, AI) oferece recursos úteis para diversos contextos de games. Alguns exemplos envolvem a inteligência de NPCs, inimigos e aliados; a orientação do jogador para alcançar seus objetivos; e situações de visão computacional onde é preciso que se "represente conhecimento" de alguma forma, seja ele tático ou estratégico para realizar alguma tarefa [4].

Já a Realidade Virtual (*Virtual Reality*, VR) age como um facilitador no processo de criação de um modelo representativo de PCG adaptável ao contexto. Os dados disponibilizados da interação do jogador imerso em um ambiente 3D podem ser analisados e servir de base para a disposição do conteúdo no espaço da cena. Com isso em vista, o contexto de restrição que a PCG remete pode ser ultrapassado, criando uma dinâmica mais ampla e trazendo a VR não apenas como ferramenta de imersão em jogos, mas como influenciadora da própria realidade imersiva. Como aponta Moreira [1], os inputs proporcionados pelo jogador agem como simplificação da edição

978-1-7281-5434-3/19/$31.00 ©2019 IEEE
DOI 10.1109/SVR.2019.00037

do cenário, que pode criar e editar sua própria cena sem a necessidade de expertise para tal.

Nesse contexto, este trabalho tem por objetivo usar a PCG em conjunto com técnicas de AI para criar um cenário virtual com suporte à VR. Um game foi desenvolvido para mostrar que a PCG pode ser utilizada como base para jogos se adaptarem automaticamente aos seus jogadores. O contexto de VR se torna útil devido as possibilidades de melhor expressar e representar as ações do jogador dentro do mundo virtual, tornando-se uma fonte de feedback cognitivo.

Para tanto, o documento está assim organizado: a Seção II apresenta abordagens relacionadas sobre PCGs orientadas por experiências, considerando qualidade de cena e síntese de padrões afetivos e cognitivos gerados durante o jogo. A Seção III mostra as ferramentas utilizadas para a criação do jogo, enquanto a Seção IV e todo o processo de desenvolvimento da AI e do VR game. Por fim, a Seção V expõe as conclusões do trabalho e os trabalhos futuros.

II. Trabalhos Relacionados

Van der Linden *et al.* [3] relatam implementações de PCG frequentemente encontradas em masmorras: um tipo específico de nível de jogo encontrado em aventuras e *role playing games*. Tal contexto é peculiarmente adequado para mostrar os benefícios do PCG devido a sua natureza aleatória e dinâmica de espaços de jogos que podem ser gerados através de algoritmos próprios. Os autores mostram métodos processuais para gerar níveis de jogo e usam a temática de masmorras como objeto de estudo para discutir os prós e contras de diferentes abordagens. Eles apresentam diversos trabalhos utilizando abordagens computacionais para a criação de masmorras e PCG, como "Celular Automata", Gramáticas, Algoritmos Genéticos e Métodos baseados em Constantes.

Jogos são vistos como um dos mais representativos exemplos de aplicações com criação e conceitualização de conteúdo. Mas também são conceituados como licitadores de complexas sínteses emocionais do usuário. Dessa forma, Yannakakis e Togelius [5] apresentam uma taxonomia para algoritmos de PCG e framework para criação de modelos computacionais de PCG baseados na experiência do jogador durante o processo interativo. Os autores apresentam diferentes tipos de abordagem para a modelagem da experiência do jogador, sendo elas:

- Modelagem Subjetiva: leva em conta apenas resultados individuais do usuário e não expressados indiretamente por fatores exteriores. O método subjetivo considera o uso de questionários simples que podem expressar de forma concisa e rápida a jogabilidade do jogador;
- Modelagem Objetiva: remete ao estado emocional do jogador e suas respostas e estímulos físicos, como expressão facial, postura e foco. Para fazer essa monitoração física é preciso hardware ou métodos apropriados para tal;
- Modelagem por Jogabilidade: considera as ações do jogador dentro do jogo proposto, tendo em vista que as próprias atitudes do jogador afetam o mundo de jogo em que ele está inserido e, da mesma forma, elicita novos padrões de foco e cognitivos. Dessa forma os padrões de jogabilidade do jogador também podem afetar o modo como o mundo se apresenta a ele.

O modelo proposto por Yannakakis e Togelius [5] procura destacar a qualidade do conteúdo do jogo. Porém a qualidade descrita por eles não se restringe somente aos modelos 3D, mas também abrange a inserção e a relação do conteúdo no contexto específico de jogo. Nesse contexto, oferecer um algoritmo de otimização que crie conteúdos divertidos, frustrantes, empolgantes ou imersivos precisa ter uma função de avaliação que reflita quanto o conteúdo específico pode contribuir para os respectivos estados afetivos do jogador durante sua exibição (reconhecidos pelos métodos de modelagem de experiência).

Já Ravenet *et al.* [6] desenvolveram um avatar robótico empático (iCat), que reconhece alguns dos estados afetivos do usuário e reage de maneira apropriada. Os autores tomaram como premissa que, pela capacidade social do avatar, os usuários estariam dispostos a manter a interação e, eventualmente, estabelecer uma relação social com ele.

Para alcançar este objetivo, buscou-se desenvolver algo que afetasse o sistema de reconhecimento capaz de detectar os estados afetivos naturais do usuário em um ambiente do mundo real. Uma vez que o avatar é capaz de reconhecer alguns dos efeitos afetivos do usuário, os autores destacam a relevância de questões como o uso do conhecimento sobre o estado do usuário para melhorar o comportamento do robô, e a mudança de perspectiva do jogador sobre o jogo.

Tendo em vista a automação e a aleatoriedade proporcionada pelas técnicas de PCG, a abordagem deste trabalho propõe definir e implementar um método de PCG baseada em AI para games imersivos. Utiliza, como base teórica, os fundamentos do trabalho de Van der Linden *et al.* [3]. Na prática, emprega-se o uso de agentes inteligentes, árvores de decisão e conceitos de jogabilidade em VR. Considera a integração de um dispositivo de visualização (HMD) e navegação (Motion Controllers) em narrativa que simula cenários, situações de jogabilidade e ambientação única a cada jogador. Sementes aleatórias e conteúdos diversos são criados para ofertar uma experiência diferenciada de jogo para cada usuário, considerando escolhas individuais durante o processo interativo.

III. Materiais e Métodos

A. Ferramentas

O jogo foi desenvolvido com recursos da game engine Unreal Engine 4 (https://www.unrealengine.com) em sua versão 4.19. Essa engine possui um sistema de programação baseado em Blueprints, onde a prática de desenvolvimento de código pode ser desviada para a concepção rápida da aplicação. Blueprints são assets (recursos ativos) com algum tipo de funcionalidade ou script relacionado. O uso desses recursos permite ao desenvolvedor atenção maior a partes específicas de implementação, e aplicar mais recursos artísticos e interativos na criação de games, simuladores ou ambientes de VR. Essa característica é importante para a modelagem de sistemas de PCG, alvo dessa proposta.

Para que os estímulos do jogador possam ser capturados de alguma forma ativa, e para que a análise da interferência no

jogo possa ser feita, foi utilizado o equipamento HTC Vive e seus controles de movimento (https://www.vive.com). Dessa forma, é possível rastrear os movimentos tanto da cabeça do jogador, como da posição e orientação de suas mãos ao longo do processo interativo. Pode-se também realizar análises a respeito de quanto tempo o jogador utiliza dentro do jogo para desempenhar alguma tarefa, e assim descobrir os lugares ou objetos em que ele demonstra mais interesse dentro da cena.

Ainda, são utilizados métodos conceituais de criação de AI como árvores de decisão e blackboards, modelos difundidos na programação de componentes inteligentes em game engines. As árvores de decisão e blackboards podem ser vistas como sistemas onde ramificações indicam tipos de tarefas específicas para resolver determinado problema, e seus resultados são divulgados em um "quadro" para que outros problemas sejam resolvidos, rápida e efetivamente, por outras ramificações.

Para esse trabalho, utilizou-se o modelo analítico de Yannakakis e Togelius [5] para adquirir feedbacks do jogador, sendo eles subjetivos, objetivos, baseados em jogabilidade, e uma opção híbrida entre estes. O intuito desse trabalho é criar, por intermédio desses métodos, formas de obter, dinamicamente, respostas do usuário ao longo do processo interativo.

IV. Desenvolvimento

Para validar a abordagem, optou-se pelo desenvolvimento de um método de PCG implementado em um VR game, capaz de gerar e inserir na cena grande quantidade de conteúdo a partir de uma cena base e do monitoramento com AI das ações do jogador. O objetivo é comprovar que as técnicas implementadas nesse estudo podem ser passíveis de utilização em jogos adaptáveis, analisando as experiências, as ações e o desempenho de cada jogador para servir de resultado empírico à mecânica do jogo. Assim, pode-se também traçar linhas de atividades que permitem saber se o jogar realmente sentiu o propósito do jogo, se o jogo se adaptou às experiências de cada indivíduo, e se o estilo do jogo mudou de acordo com o avanço do jogador.

Cabe lembrar que o objetivo do trabalho é analisar se a criação procedural de conteúdo em um VR game pode ser utilizada para adaptar o jogador a um contexto mais pessoal, a partir de um cenário base com opções temáticas. Não é alvo propor um método que defina uma jogabilidade ou crie um cenário 3D completamente do zero.

Dessa forma, primeiramente, criou-se um avatar inteligente para acompanhar o jogador no mundo virtual imersivo, auxiliá-lo durante o processo interativo, e aprender com as ações do jogador no mundo virtual. Assim, foi possível traçar linhas de interesses do jogador, permitindo a análise para criação de modelo de jogo. Uma vez que o avatar inteligente é capaz de identificar quais os estados afetivos do usuário, pode-se usar o conhecimento sobre o estado do usuário para mudar o comportamento do avatar, do cenário e da dinâmica de jogo, com o objetivo de melhorar o interesse do jogador. Leite *et al.* [7] comentam que mecanismos de jogo capazes de reconhecer e modelar o estilo de jogo e detectar o estado afetivo do usuário são marcos necessários para a personalização da experiência.

Para que o avatar auxiliasse na percepção de interações do jogador com a cena, foram utilizados dois métodos de percepção de estímulo do jogador. Um deles foi o componente AIPerception, responsável por monitorar qualquer estímulo do jogador, seja ele visual ou interativo, transformando-os em variáveis de AI. Também, para isso, foi construído um blueprint auxiliar que realizou a captação dos dados dos objetos no jogo em tempo real, os mandando em forma de mensagem para o componente de percepção do avatar. Foram traçadas linhas de atividades de exploração e interação, com base no trabalho de Yannakakis e Togelius [5].

A Unreal Engine disponibiliza para cada blueprint duas formas de programação. Uma delas está relacionada ao Event Graph, que representa funcionalidades em geral, implementadas e baseadas em eventos que podem ou não acontecer em relação ao asset de jogo. A outra forma é a implementação pelo Construction Script, onde as funções são implementadas e executadas sempre que o objeto é construído em tempo real no jogo, construções essas que ocorrem em cada mudança feita pelo desenvolvedor dentro do editor ou assim que o objeto se torna uma instância disponível no jogo.

Para tal, é gerada uma semente aleatória que interfere no índice a ser escolhido de cada característica do objeto, seja seu modelo 3D, cor, material, tamanho ou posicionamento. Dessa forma é possível criar um asset de jogo onde, por exemplo, uma árvore toma formas, tamanhos e cores diferentes de forma arbitrária. Essa natureza randômica dos objetos ainda pode ser totalmente customizada pelo designer de jogo através do editor, operando vetores de Static Meshes, conjunto de materiais ou valores que interferem nas características do objetos, como tamanho ou quantidade.

No contexto do jogo, existem elementos que são não-interagíveis ao longo do processo interativo, e que foram criados com a mesma técnica anteriormente descrita. Fazem parte desse grupo objetos de ambientação da cena, como flores, pedras, árvores, casas e luzes. Para ilustração, pode-se ver diversos tipos de árvores instanciadas de formas diferentes na Figura 1 e, usando o mesmo script, casas distintas instanciadas em diferentes posições na Figura 2.

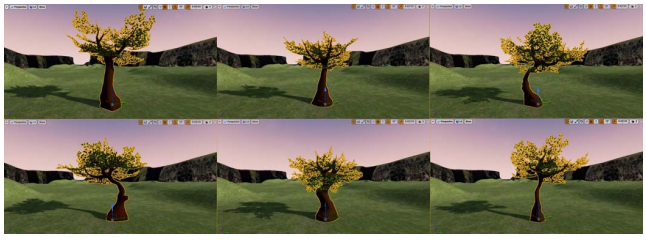

Figura 1. Diferentes tipos de árvores procedurais geradas a partir dos valores de tamanho e posição do caule no Construction Script.

Após o processo de geração, cada objeto de cena é responsável por mandar seus dados para o observador da cena, encarregado de utilizar os dados de tempo e de classe de cada um e atualizar o estado da árvore de decisão para que a mesma possa controlar como os próximos elementos do

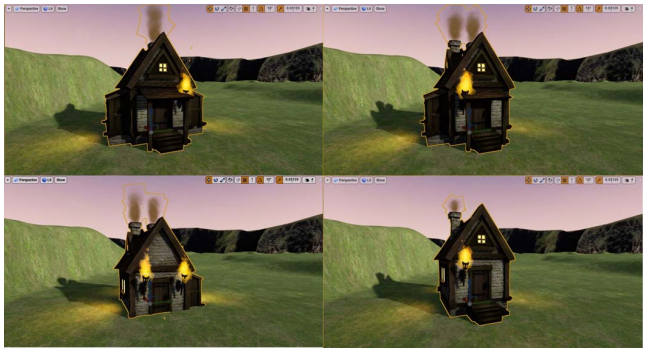

Figura 2. Diferentes tipos de casas procedurais geradas a partir dos vetores de Static Meshes no Construction Script.

cenário serão criados. As classes de objetos criados podem variar de material e texturas, assim como serem instanciadas em níveis procedurais em tempo real dentro do jogo.

Por exemplo, um jogador que tem sua interação intensificada com objetos de uma temática de bosque poderá, em certo momento, ter seu ambiente padrão de jogo modificado para um ambiente contendo mais árvores. De modo análogo, outro jogador que interagiu bem mais com as espadas disponíveis na cena base, terá um cenário que remete a algo mais destruído e com maior percepção de batalha.

É necessário que diversos objetos padrão sejam posicionados na cena inicial para que os dados de jogabilidade do jogador sejam consistentes na primeira decisão de mudança de jogo (Figura 3). As temáticas propostas para o VR game desse estudo foram "Florestal"(Figura 4) e "Combate"(Figura 5).

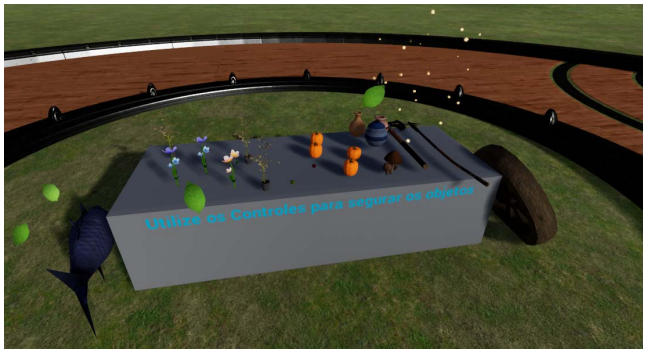

Figura 3. Cena inicial do jogo e seus objetos base.

A temática "Florestal" gira em torno de ambientes arborizados e repletos de plantas, objetos e materiais que remetem à natureza. Já a temática "Combate" leva em consideração elementos caóticos e humanizados, como cenários mais áridos e objetos cortantes, assim como um cenário mais modificado por traços humanos. Ainda, para criar uma narrativa em torno do jogo, optou-se por uma linha base de jogabilidade onde todos os jogadores podem interagir de forma uniforme. Nesse caso, um cenário de casa no campo, com conceito medieval, onde suas características e arredores podem ser definidos pelas temáticas apontadas.

O método proposto também considera que a maioria dos elementos do jogo gerados proceduralmente seja de possível interação do usuário. A ideia é permitir a experiência em um ambiente único que realce os interesses do jogador conforme a temática.

Figura 4. Cenário "Florestal", gerado a partir das interações do jogador.

Figura 5. Cenário "Combate", gerado a partir das interações do jogador.

Para conseguir mais dados sobre a jogabilidade do jogador, implementou-se também contadores de tempo de interação, associados a cada objeto de cena. Assim, pode-se de forma cumulativa saber quanto tempo o usuário interagiu com cada elemento de cena, ativando a contagem de tempo no momento da seleção, e desativando na liberação deste à cena. Definiu-se para o método proposto que quanto maior fosse o tempo de interação com um objeto ou uma mesma classe de objetos, maior o interesse do jogador naquele tipo de contexto. Tais dados de interesse guiados por tempo também foram aplicados como parâmetros de alimentação da AI responsável pela criação procedural e do andamento do jogo.

Além dos contadores, os seguintes métodos foram utilizados para adquirir dados sobre o jogador com base no método de modelagem de experiência de Yannakakis e Togelius [5]:

- Questionário virtual (uso de GUIs 3D): para que se apresentasse de forma simples e direta ao jogador, utilizou-se questionário baseado em imagens, com duas perguntas

que buscavam representar o interesse do jogador no início do jogo;
- Interação com objetos com Mão Virtual: definiu-se pelo uso da técnica de mão virtual para representar as mãos do usuário, mapeadas a partir da posição real dos Motion Controllers do HTC Vive. Assim, o jogador poderia pegar e segurar os objetos virtuais de forma similar à interação real, e o jogo poderia ter acesso a posição e orientação dos controles;
- Visualização de objetos com HMD: para identificar o foco do jogador dentro do jogo, utilizou-se de um vetor frontal localizado na frente da posição virtual do Headset do HTC VIVE. Dessa forma, uma linha é traçada em frente à visão do jogador, sendo assim possível identificar locais de foco e objetos para os quais o jogador olha durante a experiência do jogo.

Para que o tempo de interação de cada objeto possa ser usado como parâmetro, deve-se levar em conta a classe do objeto em questão e o tempo em que ele passou interagindo com o usuário. Para isso, criou-se uma classe de objetos (InteractableObject) responsável em monitorar as ações e o tempos de manipulação ou visualização de cada objeto, associando-os por um identificador deste na cena.

Os triggers de ativação dos contadores levaram em consideração a proximidade da mão virtual em relação aos objetos. Dessa forma, com os triggers traseiros do Motion Controller foi possível simular a ação de segurar objetos, informando a anexação (grab) do modelo 3D da cena na mão virtual. Por essa questão, os tempos só são processados depois que o jogador solta os triggers e o objeto é liberado (release) de volta à cena.

Para que novos objetos de cena sejam instanciados pela AI e façam sentido nesta, foram implementados diferentes níveis procedurais para assegurar a geração aleatória na cena dentro do contexto do jogo. Esses níveis garantem que os objetos estejam dispostos de forma coerente para que o propósito de jogo seja cumprido, seja ele apenas para interação livre ou associado a uma tarefa específica do jogador.

Para tanto, a Unreal Engine disponibiliza de recursos de "Level Streaming", que são sub-níveis que podem ser instanciados dentro da cena principal usando blueprints, sem ocupar ou fazer parte do contexto da cena principal. É possível exemplificar pensando em uma cena hipotética de uma casa que possui como ambiente principal uma sala de estar, e onde o conteúdo dos demais cômodos são separados por portas. Dessa forma, esses cômodos podem ser tratados como sub-níveis da cena, e apenas apresentados ao jogador se ele abrir a porta. Isso otimiza o processo de renderização e ocupação de memória, evitando que cenas muito grandes ou complexas demorem a carregar ou prejudiquem a interação no espaço.

Esses sub-níveis procedurais foram implementados com "Level Streaming" e separados por temáticas, sendo que sua renderização e posicionamento são controlados pela AI com base nos dados do usuário, adquiridos durante a realização da tarefa na cena.

V. Conclusão

Este estudo apresentou a definição de um método de PCG baseado em AI por meio da implementação de um VR game. Para tanto, técnicas de PCG baseada em agentes e árvores de decisão foram utilizadas em uma aplicação de VR com suporte ao HTC Vive.

Em relação à AI, cabe destacar que ela cumpriu com as expectativas de funcionamento de acordo com a dinâmica do jogo, levando em consideração o funcionamento a ela proposto. Porém, é possível fazer melhorias nos algoritmos da aplicação, no sentido de oferecer melhor responsividade das ações do jogador. Além disso, pode-se desenvolver um sistema de subníveis pré-determinados que permita ao programador ou designer reusar em novos níveis.

Como trabalhos futuros, sugere-se primeiramente uma avaliação preliminar do game desenvolvido, para validar a experiência diferenciada de jogo para cada usuário, considerando suas escolhas individuais durante o processo interativo.

É possível criar um sistema de recomendação de maior precisão aos estímulos dos usuários, assim como uma narrativa mais delineada a um jogo. Também pode-se utilizar implementações jogáveis, como mini-games, perguntas e conteúdo jogável interativo, de forma a alimentar ainda mais a AI com outros tipos de variáveis, como tempo de resolução de problemas e tipos de conteúdo (puzzles, FPS ou corrida, por exemplo).

Também sugere-se avançar para um jogo completo e disponível à comunidade, onde jogadores, de forma cooperativa ou até mesmo on-line, possam criar ou fazer upload de seu próprio conteúdo. Eles poderiam categorizá-lo no contexto do jogo e gerar de forma dinâmica mapas para toda a comunidade de jogo. Dessa forma, pode-se generalizar ainda mais a experiência, tornando-a mais complexa em relação a todos os contextos e perfis de jogadores.

Referências

[1] J. F. M. Moreira, "Mixed interactive/procedural content creation in immersive vr environments," Master's thesis, Mestrado Integrado em Engenharia Informática e Computação, Universidade do Porto, 2018.

[2] J. Fingas, "Here's how 'Minecraft' creates its gigantic worlds," https://www.engadget.com/2015/03/04/how-minecraft-worlds-are-made/, 2015.

[3] R. Van der Linden, R. Lopes, and R. Bidarra, "Procedural generation of dungeons," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 6, no. 1, pp. 78–89, 2014.

[4] G. N. Yannakakis and J. Togelius, *Artificial intelligence and games*. Springer, 2018, vol. 2.

[5] ——, "Experience-driven procedural content generation," *IEEE Transactions on Affective Computing*, vol. 2, no. 3, pp. 147–161, 2011.

[6] B. Ravenet, F. Pecune, M. Chollet, and C. Pelachaud, "Emotion and attitude modeling for non-player characters," in *Emotion in Games*. Springer, 2016, pp. 139–154.

[7] I. Leite, A. Pereira, S. Mascarenhas, G. Castellano, C. Martinho, R. Prada, and A. Paiva, "Closing the loop: from affect recognition to empathic interaction," in *Proceedings of the 3rd international workshop on Affective interaction in natural environments*. ACM, 2010, pp. 43–48.