Revista de Sistemas de Informação da FSMA
n. 18 (2017) pp. 43-57

SISTEMAS DE INFORMAÇÃO
http://www.fsma.edu.br/si/sistemas.html

TRILHA PRINCIPAL

Geração Procedural de Conteúdos para Jogos Digitais: Conceitos e Trabalhos Relacionados

Julian R. H. Mariño , *Doutorando em Ciência de Computação e Matemática Computacional, Instituto de Ciências Matemáticas e Computação (ICMC) - Universidade de São Paulo(USP)*,
Leonardo T. Pereira , *Mestrando em Ciência de Computação e Matemática Computacional, Instituto de Ciências Matemáticas e Computação (ICMC) - Universidade de São Paulo(USP)* e
Claudio F. M. Toledo , *Prof. Dr., Departamento de Sistemas de Computação Instituto de Ciências Matemáticas e Computação (ICMC) - Universidade de São Paulo(USP)*

***Resumo*—O mercado de jogos digitais cresce a cada ano e o processo de desenvolvimento desses jogos se torna cada vez mais complexo. Assim, a escalabilidade na geração de conteúdos pode demandar o trabalho de uma equipe de centenas de pessoas. A Geração Procedural de Conteúdo (GPC) surge como uma alternativa para reduzir custos e acelerar o processo de produção dos jogos, criando conteúdos de maneira automática ou semi-automática. O presente artigo apresenta alguns conceitos e revisa trabalhos desenvolvidos em GPC, procurando fornecer um horizonte inicial para aqueles interessados em conhecer e se aprofundar em GPC para jogos digitais.**

***Palavras-chave*—Jogos Digitais, Geração Procedural de Conteúdos, Algoritmos de Busca.**

Procedural Content Generation: Concepts and Related Works

***Abstract*—The digital games market is growing every year and game development is becoming increasingly complex. Thus, scalability in content generation may require the work of a team with hundreds of people. Procedural Content Generation (PCG) comes as an alternative to decrease costs and accelerate the process of game production by creating content automatically or semi-automatically. This article presents some concepts and reviews works developed in PCG, aiming to provide a starting point for those interested in learning and going deeper in the subject of PCG for digital games.**

***Index Terms*—Digital Games, Procedural Content Generation, Search Algorithms**

I. INTRODUÇÃO

O ANTROPÓLOGO Johan Huizinga [1] descreve jogo como mais do que um mero fenômeno fisiológico ou um reflexo psicológico, mas um conceito que transcende o de uma atividade puramente física ou puramente biológica. Segundo o mesmo autor, o jogo transcende as necessidades imediatas da vida e dá sentido à ação.

A criação dos computadores e dos *displays* gráficos levou ao aparecimento de uma nova modalidade de jogos: os jogos digitais. Atualmente, esses representam um dos maiores e mais rentáveis mercados internacionais com receita mundial de $91,8 bi em 2015, sendo $1.5 bi somente no Brasil, que foi o 11º maior mercado do ramo no mesmo ano[2][3].

Atualmente, existem mais de 2,1 bilhões de jogadores no mundo, sendo que 33,6 milhões são brasileiros [4]. No ano de 2015, apenas para a plataforma de jogos Steam foram lançados quase 3500 jogos [5]. Além desses, mais de 1000 jogos foram lançados no mesmo ano para os consoles de mesa e portáteis [6]. O mercado destinado para celulares é o que possui maior volume de publicação. Somente em janeiro de 2016, 19.130 jogos foram submetidos para lançamento na loja de aplicativos do iTunes [7] [8].

O crescimento do número de jogadores e o aumento na complexidade dos jogos digitais trouxeram um problema de escalabilidade na geração de conteúdo [9] [10]. A produção de jogos com alta qualidade demanda o trabalho de uma equipe de centenas de pessoas que inclui artistas, game designers, programadores e engenheiros de áudio [11] [12]. A maioria desses profissionais trabalha na geração de conteúdos, que se tornou uma atividade com impacto relevante no custo e tempo de produção [13]. Por exemplo, jogos com maiores orçamentos de produção e *marketing*, chamados de *Triple-A*, podem custar U$200 mi apenas para serem desenvolvidos [14].

Uma alternativa para reduzir custos e acelerar o processo de produção dos jogos é a Geração Procedural de Conteúdo (GPC) ou, em inglês, *Procedural Content Generation - PCG*. Segundo [15], PCG utiliza algoritmos para gerar conteúdos de maneira automática ou semi-automática. Assim, os conteúdos criados podem ser utilizados como ponto de partida por *game designers* no processo de elaboração da versão final do jogo. Outra vantagem é que tais conteúdos também podem ser gerados em tempo real para jogos, por exemplo, através de sua adaptação às habilidades ou perfil do jogador. Assim, a coleta de informações sobre o comportamento do mesmo em tempo real pode levar à geração de conteúdos específi-

cos naquela etapa. Esse tipo de abordagem pode aumentar o entretenimento do usuário, permitindo diferentes níveis de experiência a cada nova partida [16].

O presente artigo discorre sobre a GPC, apresentando alguns conceitos e revisando trabalhos desenvolvidos neste tema. Uma revisão bibliográfica completa foge ao escopo deste trabalho e, para isso, recomendamos a leitura de [17][15]. Nosso objetivo principal é apresentar um texto que sirva de horizonte inicial para aqueles interessados em conhecer e se aprofundar em GPC para jogos digitais e, a partir disso, pode-se dizer que a maior contribuição deste trabalho é servir como introdução de estudantes e profissionais que queiram começar a se familiarizar com a área de GPC.

O artigo está organizado como se segue. A seção II apresenta conceitos evolvendo a GPC. Em seguida, trabalhos utilizando algoritmos baseados em busca são descritos na seção III. Trabalhos relacionados à técnicas de avaliação dos conteúdos gerados proceduralmente são revisados na seção IV. A seção V apresenta trabalhos utilizando GPC em determinadas áreas de aplicação em jogos. Por último, as conclusões obtidas seguem na seção VI.

II. Motivação e Conteúdos Gerados

Um motivador para o uso de GPC foi a capacidade de memória restrita dos computadores no início da década de 80. Tal restrição limitava a quantidade de conteúdo capaz de ser gerada, levando os desenvolvedores a inovarem, criando conteúdos dinamicamente [18]. Esse foi o caso do jogo *Elite* [19] cujas 8 galáxias, compostas por planetas com características distintas, eram geradas dinamicamente uma vez que não havia como armazenar todas em memória.

Apesar do aumento na capacidade de armazenamento dos computadores ao decorrer dos anos, a geração dinâmica ou automática de conteúdos continuou sendo utilizada. Porém, agora para permitir ao jogador ter experiências diferentes a cada nova partida (*replayability*) [16]. Por exemplo, o jogo de plataforma 2D *Spelunky* [20] utiliza GPC para criar automaticamente variações de níveis.

Os autores em [21] estabelecem diversos conceitos relacionados à GPC. De forma geral, um sistema de GPC é formado por um ou mais algoritmos que geram conteúdos. Ele ponde funcionar como uma ferramenta de *design* de jogos assistida por técnicas de Inteligência Artificial (IA). Tal sistema auxilia desenvolvedores, projetistas e artistas a melhorar o conteúdo relacionado ao jogo.

O objetivo principal de um sistema de GPC é a criação de conteúdo a partir de determinados objetivos que atendam requisitos específicos de qualidade. Por exemplo, se o objetivo principal é a criação de conteúdos visualmente agradáveis, um conteúdo resultante será considerado satisfatório caso possua boa estética visual. Os conteúdos a serem gerados em um jogo digital podem ser níveis, histórias, regras do jogo, personagens, texturas, armas, entre outros [21].

De acordo com [22], um nível pode ser definido como um ”contêiner para jogabilidade”, ou seja, representa o ambiente do jogo com todos os elementos nele contidos. A geração de níveis é um dos problemas mais clássicos tratados pela GPC e concentra boa parte das pesquisas recentes [23].

Os autores em [24] aplicam uma busca heurística para selecionar a posição de objetos em uma versão do jogo *Super Mario Bros.* (SMB) chamada *Infinite Mario Bros.* (IMB). O trabalho consegue validar a hipótese de que teorias de *design* gráfico, normalmente utilizadas em outras áreas, podem ser aplicadas para guiar a geração de níveis de modo a apresentar uma visualização gráfica agradável. A Figura 1 mostra um exemplo de uma fase gerada por este sistema.

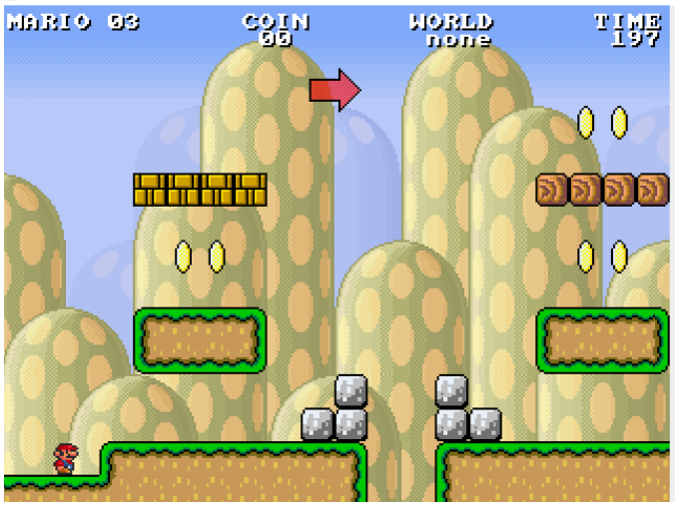


Fig. 1: Fase gerada por GPC para o jogo Infinite Mario Bros, descrito em [24]. Note que as posições dos blocos e as plataformas são determinadas por um algoritmo de busca heurística.

Os autores em [25] aplicam uma abordagem baseada em busca (*search-based approach*) para gerar níveis com alta qualidade e diversidade para jogos de estratégia. O trabalho propõe o uso da GPC como uma ferramenta de IA capaz de auxiliar o *game designer* ao oferecer vários níveis de boa qualidade com diferenças relevantes entre eles. A ferramenta proposta permite uma interação entre o sistema e o designer, na qual o sistema pode sugerir alternativas de níveis a partir de outros, previamente desenvolvidos pelo designer. O principal objetivo é favorecer a diversidade das fases criadas sem perda de qualidade. Algoritmos evolutivos são usados para esse propósito.

O trabalho descrito em [23] introduz o conceito de um sistema de GPC de propósito geral, através de um ambiente (*framework*) base. Este permite que pesquisadores apliquem a GPC para criação de níveis em diferentes jogos digitais. Para que seja possível a geração de níveis diferentes, o ambiente permite que os sistemas geradores interpretem uma descrição do jogo desejado, incluindo suas restrições. Por exemplo, caso o objetivo seja gerar níveis para um jogo do tipo *Pac Man* [26], uma das restrições será que o jogo só deverá ser finalizado quando todos os inimigos estiverem mortos.

Um objetivo comum, embora com um nível de dificuldade alto na geração de níveis, é a maximização da diversão que o jogador pode experimentar. Os autores em [27], apresentam um esquema de seleção e calibragem de parâmetros de jogo. Algoritmos de aprendizado por reforço buscam a melhor combinação desses parâmetros para que a diversão seja maximizada. Experimentos com um jogo de corrida de motos, permitiram aos jogadores avaliarem o conteúdo segundo sua diversão. Ao final dessas avaliações, os desenvolvedores contaram com um *ranking* de configurações de parâmetros.

A utilização de conteúdos previamente avaliados por seres humanos, visando uma posterior geração semiautomática do conteúdo, é aplicada também em [28]. O objetivo é maximizar a diversão durante o jogo, conforme será apresentado em detalhe na seção V-A.

A narrativa em jogos é outro tipo de conteúdo a ser gerado que permite contextualizar a ação ao ambiente do jogo [21]. Um exemplo clássico é o jogo de tiro em primeira pessoa (*first-person shooter*) *Doom* [29] que narra a falha em um experimento que leva à abertura das portas do inferno. O objetivo principal da narrativa é dar um significado em forma de objetivo, ajustando-o ao tema geral do jogo. Assim, GPC pode ser utilizada para criar conteúdos relacionados à narrativa, focando em permitir aos jogadores alterar dinamicamente tal narrativa através de suas ações.

Os autores em [30] exploram uma narrativa flexível ao introduzir uma linguagem para os desenvolvedores. Esta é baseada em normas e preferências que permitem a construção de narrativas maleáveis, utilizando um agente inteligente. Tais preferências são aplicadas para descrever o tipo de conteúdo visualizado pelo jogador, enquanto as normas são regras específicas e pré-definidas para o jogo.

O sistema PACE [31] é implementado e testado em [32] para gerar uma narrativa que interage com o jogador. O sistema busca mantê-lo em sequências emocionais previamente fornecidas pelo autor, através de predições baseadas na resposta do jogador a um evento específico do jogo. Logo, o próximo segmento da narrativa é aquele que se encaixa melhor dentro da serie emocional definida pelo autor. Por exemplo, num dado momento do jogo, deseja-se que o jogador seja estimulado a continuar uma sequência de ações mais relacionadas ao ataque do que defesa. Assim, o sistema escolherá sequência de eventos da narrativa que estimulem esse comportamento no jogador.

Os autores em [33] aplicam um planejamento automático para escolher a sequência de eventos que melhor se adapte à intenção original do autor, depois que o jogador executa uma ação. Isso permite ao jogador modificar eventos da história original sem que o objetivo pré-definido pelos autores do jogo seja violado.

Comentários esportivos são um tipo especial de narrativa fundamental para jogos como futebol, beisebol, basquete, entre outros. O sistema SCoReS é proposto em [34] para produzir comentários adequados baseados no estado atual do jogo. O sistema é treinado *offline*, utilizando técnicas de aprendizado de máquina, para mapear adequadamente o estado atual em comentários de entretenimento. Tal sistema é aplicado a jogos de beisebol com comentários reproduzidos tanto durante o jogo como nos intervalos.

A música é outro tipo de conteúdo que pode ser fundamental na avaliação da qualidade de um jogo, motivo pelo qual tem sido também objeto de estudo na área. A geração procedural deste conteúdo consiste no uso de técnicas computacionais para a criação automática ou semiautomática da música que acompanha um jogo [35].

Os autores em [36] utilizam técnicas baseadas em busca impulsionada pela experiência (*experience-driven PCG*) para compor músicas que se adaptem ao estilo de jogo. Algoritmos evolutivos foram responsáveis pela busca. Em [37] os autores mostram como a criação adaptativa da música, seguindo o comportamento dos personagens do jogo, faz com que as emoções e expressões demonstradas por eles sejam mais realistas.

Uma interação ou relação entre tipos específicos de conteúdos pode ser utilizada no processo de GPC. Por exemplo, os autores em [38] usam música como uma ferramenta auxiliar na geração do comportamento de armas durante o jogo. Essa abordagem estabelece uma relação entre dois conteúdos específicos, música e armas, onde a música pode ser mapeada em armas durante o jogo. Portanto, mudanças na música levam a mudanças nas armas. A forma como o jogador utiliza tais armas também pode influenciar no processo de mapeamento música-arma.

Os autores em [39] apresentam um trabalho no qual a música é o ponto inicial para criação de outra classe de conteúdo. Os autores introduzem *BeatTheBeat*, um jogo para dispositivos móveis no qual técnicas de IA são utilizadas para extrair características a partir de músicas. As características foram usadas para a geração automática de níveis. As músicas são representadas por vetores de características de várias dimensões, permitindo que mapas auto-organizáveis sejam utilizados. Nesse caso, os mapas funcionam como uma rede neural que mapeia um conteúdo específico, representado por várias dimensões, em um mapa de apenas duas dimensões.

O estudo conduzido em [40] relaciona a forma como as emoções dos personagens do jogo (não-jogadores) podem ser demonstradas de maneira crível para o jogador. Um modelo computacional baseado em modelos psicológicos de emoções é proposto para controlar as reações desse tipo de personagem.

Outro tópico de grande interesse na área de GPC é a geração de mapas. Devido à popularidade de jogos de calabouços e cavernas desde a década de 80 até hoje, a criação de mapas para este gênero de jogo se torna um problema relevante. Um algoritmo baseado em geração recursiva de terrenos é proposto em [41], no qual a validação de tamanhos e correção na coesão dos mapas são usados para criação de mapas uniformes.

Uma ferramenta de assistência ao designer na geração de mapas para jogos de calabouços é apresentada em [42]. Neste trabalho, o designer apresenta esboços de possíveis mapas desejados a partir dos quais o sistema avalia a

jogabilidade e viabilidade, criando novos mapas.

TABELA I: Trabalhos baseados em Técnicas de GPC.

Niveis	Narrativa	Musica	Armas	Mapas de calaboucos	Comentarios esportivos	Emoções
[24], [25], [23], [27], [28], [39]						
	[30], [31], [32], [33]					
		[36], [37]				
			[38]			
				[41], [42]		
					[34]	
						[40]

III. Algoritmos Baseados em Busca para GPC

A presente seção descreve a abordagem mais empregada na literatura para GPC: algoritmos baseados em busca. Nesse contexto, o problema de GPC é tratado como um problema de otimização cujo objetivo é encontrar o valor ótimo correspondente a uma configuração específica de conteúdo, segundo [43].

Diferentes tipos de algoritmos baseados em busca podem ser aplicados à GPC: arrefecimento simulado, otimização de enxames de partículas, algoritmos evolutivos, entre outros. Trata-se de metaheurísticas cuja aplicação é recomendada quando a busca é realizada em um espaço extenso de configurações possíveis, ou o tempo disponível para encontrar um bom conteúdo é limitado. Metaheurísticas não garantem a obtenção da solução ótima para o problema, mas costumam retornar boas soluções dentro de um tempo computacional razoável quando bem formuladas [44].

Independente do tipo de técnica de busca a ser aplicada na GPC, em geral, há dois componentes principais a serem considerados: representação do conteúdo e função de avaliação. Esses componentes serão apresentados nas seções que seguem.

A. Representação do Conteúdo

A representação do conteúdo contém as informações necessárias para a devida construção do mesmo no espaço do problema [45], podendo ser realizada através de uma codificação direta ou indireta. A direta apresenta informações que permitem mapear partes específicas do conteúdo. Por exemplo, um algoritmo evolutivo é aplicado em [46] para criar níveis no jogo Super Mario Bros. O método usa um vetor de segmentos verticais contendo os objetos do jogo para representar determinado nível. Assim, um nível é gerado ligando todos os segmentos codificados no vetor.

A codificação indireta demanda um mapeamento não linear do conteúdo, o que pode aumentar a complexidade envolvida no processo de decodificação da representação [47]. Por exemplo, os autores em [48] utilizam uma codificação indireta para representar níveis do jogo Angry Birds [49] em um Algoritmo Genético (AG). A codificação é ilustrada na Figura 2, onde o nível é representado pelo número de pássaros e de blocos sobrepostos. Cada bloco é codificado como duas coordenadas (x,y) onde x representa o tipo de bloco e y assume os valores 0 ou 1, no qual 1 indica que o bloco é duplicado. Um exemplo de bloco duplicado pode ser visto no primeiro bloco da pilha 2 (*stack 2*).

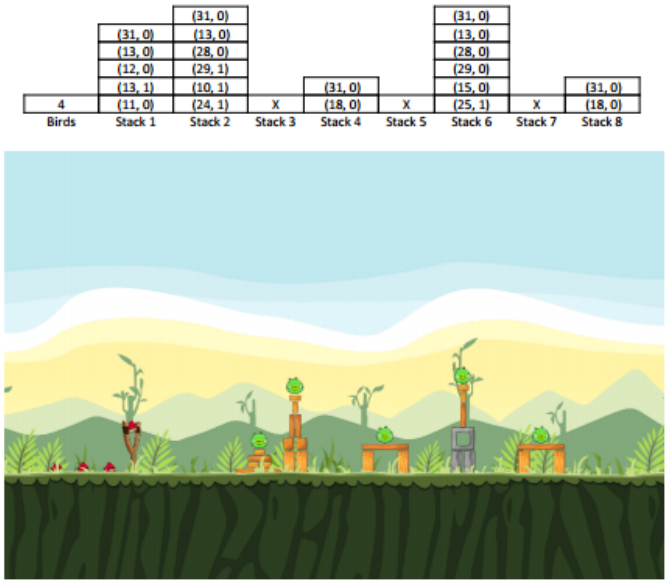


Fig. 2: Representação indireta de um nível de Angry Birds descrita em [48]. Note que a sequência de pilhas na representação indireta na parte superior da figura representa a sequência de elementos do nível gerado na parte inferior. Por exemplo, o numero 4 no primeiro item de pássaros (*Birds*) na representação indireta representa os 4 pássaros colocados no início do nível real.

Os autores em [50] e [15] destacam alguns pontos relevantes sobre a escolha da codificação. Uma codificação indireta deve buscar garantir que todo o espaço de busca por conteúdos seja alcançável. Isso nem sempre é fácil de se garantir e certas regiões podem não ser mapeadas. Essa tarefa pode ser mais simples com uma codificação direta. Além disso, pequenas alterações na codificação devem resultar em pequenas alterações no conteúdo. Isso, em geral, leva a uma pequena alteração também no valor da função de aptidão.

A codificação utilizada também precisa lidar com as restrições relativas à geração dos conteúdos. Por exemplo, considere um jogo de plataforma: é fundamental executar uma trajetória entre o início e o fim de um nível. Todavia, uma representação do nível, ao ser decodificada, pode levar à geração de um nível infactível. Uma solução para tal problema é desconsiderar tais conteúdos e suas respectivas codificações, porém isso pode impactar negativamente o processo de busca [51]. Outra alternativa é a penalização da codificação no cálculo da função de avaliação [52] o que pode levar o método a avaliar uma quantidade elevada de representações infactíveis. Um algoritmo de reparo pode ser aplicado para corrigir infactibilidades durante o processo de codificação ou decodificação da representação.

Uma abordagem mais elaborada consiste em avaliar de forma distinta codificações factíveis e infactíveis, como proposto no algoritmo *FI-2Pop GA* [53] e suas variantes apresentadas em [54]. Trata-se de um método que aplica

algoritmos genéticos, cuja primeira população é formada apenas por codificações factíveis e sua função de avaliação prioriza o tipo de conteúdo desejado. As codificações infactíveis estão na segunda população, cuja função de avaliação prioriza o nível de factibilidade dos conteúdos. Assim, codificações factíveis na segunda população migram para a primeira e as infactíveis geradas na primeira população migram para a segunda.

B. Função de Avaliação

Definir uma função de avaliação demanda decidir qual aspecto do jogo precisa ser otimizado e como isso pode ser formalizado. Por exemplo, um algoritmo baseado em busca pode ser definido para otimizar a diversão do jogo. Desta forma, a função de avaliação deve ser capaz de identificar o quanto um determinado conteúdo gerado contribui para a sensação de diversão do jogador durante o jogo. De forma geral, as funções de avaliação podem ser classificadas em três tipos: direta, interativa e baseada em simulação [55].

Na avaliação direta, apenas as características do conteúdo são utilizadas para o cálculo da aptidão, sendo que as funções para esse cálculo podem estar baseadas em teorias (*theory-driven*) ou em dados (*data-driven*) [15]. No primeiro caso, o desenvolvedor da função de avaliação é guiado por teorias qualitativas de *game design* sobre geração de conteúdo e avaliação de experiência do jogador. No segundo caso, são utilizados dados coletados sobre o efeito de diversos conteúdos na experiência do jogador.

Diferentes exemplos de funções de avaliação diretas são apresentadas em [56], no qual um AG co-evolutivo é aplicado para gerar jogos inteiros no estilo *arcade*. O algoritmo explora três aspectos do jogo: mapa, posição dos inimigos e um conjunto de regras. Cada um deles possui uma função de avaliação direta própria. Por exemplo, a qualidade da disposição dos inimigos é calculada utilizando a esparsidade e o volume da matriz que a representa. Um nível será considerado esparso quando a distância media entre dois inimigos da tela é alta. Dado um mapa m com n inimigos, essas métricas podem ser calculadas da seguinte forma:

$$esparsidade(m) = 2 * (\sum_{i=1}^{n}\sum_{j=1}^{n} dist(i,j)) * \frac{1}{n*n-1} \quad (1)$$

$$vol(m) = |vermelhos(m)| + |azuis(m)| + |verdes(m)| \quad (2)$$

Sendo que $dist(i,j)$ é a distância euclidiana entre os inimigos i e j, $|vermelhos(m)|$ é a quantidade de inimigos da cor vermelha, $|azuis(m)|$ é a quantidade de inimigos da cor azul e $|verdes(m)|$ é a quantidade de inimigos da cor verde. Deve-se ressaltar que a função de avaliação da disposição dos inimigos é calculada utilizando apenas o estado da matriz que representa a disposição. Por isso, este tipo de função é dita como avaliação de conteúdo direta.

A função de avaliação interativa permite que o conteúdo seja avaliado por um jogador (ou *designer*) humano durante o processo de geração [57]. Os dados sobre a avaliação são captados de maneira explícita ou implícita. As avaliações explícitas podem utilizar questionários, enquanto as implícitas buscam obter algumas métricas, durante o jogo, que reflitam a experiência do jogador. Por exemplo, calcular quantas interações o usuário teve com um determinado tipo de conteúdo, o momento em que o jogador saiu do jogo, ou a pressão utilizada para apertar um botão [15].

Os autores em [51] introduzem um *framework* para geração de pistas para jogos de corrida que são avaliadas interativamente pelos jogadores. Esse *framework* é composto por duas partes: uma interface *web*, responsável pela interação com o usuário para avaliação das pistas, e um servidor que aplica os operadores evolutivos de seleção, recombinação e mutação. O processo evolutivo passa a ser guiado pelo usuário, ou seja, os operadores da evolução são executados pelo servidor apenas depois que se avaliam as pistas da população corrente e pressiona-se o botão *"Evolve!"*.

A definição da qualidade de um conteúdo com base apenas em suas características não é uma tarefa simples [58]. Assim, o valor agregado ao jogo por um conteúdo pode ser avaliado usando simulação. Nesse caso, o conteúdo é inserido no jogo e um agente inteligente artificial é utilizado para simular um jogador real. Nesse caso, características são extraídas durante a execução da simulação para avaliação do conteúdo [15].

Esse tipo de avaliação pode ser feita de maneira dinâmica ou estática [50]. A avaliação é dita estática quando o agente utilizado na simulação não é alterado durante o jogo. Por exemplo, *Cut the Rope*, um jogo de quebra-cabeças baseados em física, no qual os autores de [50] aplicam um agente capaz de resolver tais quebra-cabeças. Esse agente é integrado a um gerador de níveis chamado *Ropossum* [59] que verifica, por meio de simulações, se os níveis são jogáveis ou não.

Numa avaliação dinâmica, o agente é alterado durante a execução da simulação e a função de avaliação passa a levar em conta tais alterações. Por exemplo, um agente pode ser implementado utilizando algum algoritmo de aprendizado, dessa forma a aptidão do conteúdo seria relativa à quantidade de aprendizado que o agente obteve na simulação.

Uma função de avaliação baseada em simulação é proposta em [60] para gerar níveis do jogo Angry Birds. Um AG evolui os conteúdos gerados, e cada conteúdo é avaliado através de simulações que medem o nível de estabilidade das estruturas empilhadas nos níveis. Os níveis mais estáveis apresentarão melhor valor de *fitness*. A Figura 3 mostra um exemplo de um nível gerado neste trabalho.

Um sistema de GPC orientada por experiência (*experience-driven PCG*) é proposta em [61]. Uma combinação de vários algoritmos evolutivos é aplicada para a geração de mapas em um jogo de ação e tiro. O objetivo principal é a geração de conteúdos personalizados para cada jogador de acordo com suas preferências. Para possibilitar determinada personalização, o sistema funciona de forma *online*, aprendendo as preferências dos jogadores

Fig. 3: Melhor nível obtido após 200 interações do algoritmo evolutivo proposto em [60] para geração de níveis com estruturas estáveis. Nesse experimento, um número máximo de 20 blocos foi considerado.

durante o jogo.

Jogos de estratégia em tempo real são um desafio interessante para pesquisas de IA. Por exemplo, um algoritmo evolutivo multi-objetivo foi proposto em [62] para a geração de mapas no jogo de estratégia *Starcraft* [63]. Um conjunto de funções de avaliação é definido de acordo com várias características desejáveis neste tipo de jogo: jogabilidade, equidade, diferenciação de habilidade e interesse.

A *jogabilidade* é definida como a possibilidade de que todas as ações possam ser executadas normalmente no mapa. A *equidade* significa que todos os jogadores devem ter a mesma chance de ganhar. Uma *diferenciação de habilidade* está relacionada à capacidade do mapa de favorecer aos jogadores com melhores estratégias de jogo, enquanto o *interesse* representa a variedade estrutural dos níveis gerados. As funções estabelecidas em [62] avaliam os conteúdos através de simulações.

Nos jogos de estratégia em tempo real, é comum encontrar mecânicas baseadas em batalhas de exércitos nas quais o jogador deverá produzir de forma eficiente um exército de batalha. Um AG é utilizado em [64] para apoiar a geração desses exércitos. Como as formações deles podem ser baseadas num esquema defensivo ou ofensivo, a função de avaliação usada nesta proposta tenta maximizar o desempenho ofensivo na geração das formações para tentar conseguir uma vitória em todos os casos. Foram utilizadas três tipos de representações para os cromossomos: segundo o equipamento do herói, segundo um conceito do jogo chamado de *Unassigned Trait Points (UTPs)*, e segundo a organização dos soldados na formação.

Um algoritmo para geração de níveis para o jogo Angry Birds é descrito em [65], no qual um AG procura por níveis com dificuldade compatível às necessidades individuais dos jogadores. O ajuste de dificuldade é controlado pela função de aptidão e, para análises da estabilidade das estruturas criadas, é utilizado um conjunto de cálculos baseado no conceito de *extended rectangle algebra* descrito em [66].

TABELA II: Trabalhos baseados em Técnicas de Busca.

Representação de Conteúdo		Função de Avaliação		
Direta	Indireta	Direta	Iterativa	Simulação
[46]				
	[48], [52], [53], [54], [51], [60]			
		[56]		
			[51], [57], [61]	
				[50], [59], [60], [62], [64], [58]

IV. TÉCNICAS DE AVALIAÇÃO

Algoritmos baseados em busca, conforme descrito na seção anterior, são a principal técnica empregada na GPC. Todavia, surge a questão de como avaliar se o conteúdo final retornado por tal técnica satisfaz de fato os requisitos desejados pelo *game designer*. Na literatura, diferentes formas de avaliar o conteúdo final gerado são propostas. O objetivo é determinar se ele é satisfatório segundo alguma característica desejada, como diversão, estética visual, dificuldade ou diversidade das fases geradas. Nesta seção, descrevemos trabalhos que se valem de diferentes técnicas para avaliar o conteúdo final gerado.

A. MÉTRICAS COMPUTACIONAIS

Em muitos trabalhos, são utilizadas métricas computacionais para avaliar automaticamente o conteúdo. Muitas dessas métricas tentam avaliar o conteúdo segundo um conceito chamado de *expressividade* [67] [68], proposto como protocolo de avaliação da variedade estrutural de níveis que um sistema consegue gerar.

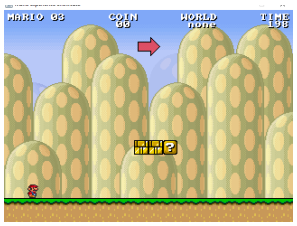
Nesse contexto, as métricas mais comuns são linearidade, leniência e densidade. A métrica de *linearidade* avalia a linearidade de um nível em função das diferenças de altura entre alguns objetos. A *leniência* calcula valores em função da dificuldade daquele nível. A *densidade* é determinada segundo o número de objetos sobrepostos em um nível.

Os autores em [57] apresentam *Tanagra*, um sistema que consegue criar, de forma semi-automática, fases de um jogo de plataformas cujas fases são criadas a partir de restrições definidas pelo *designer*. Em *Tanagra*, as fases obtidas são avaliadas de acordo com métricas definidas em [67].

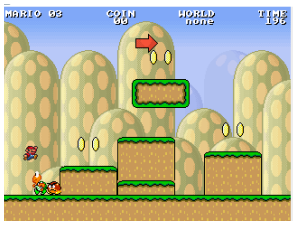
Um jogo de plataforma é apresentado em [69], no qual pequenos trechos de conteúdo (*grupos de ritmos*) são unidos para formar níveis inteiros. Para isso, alguns parâmetros pré-definidos pelo *designer* são levados em consideração. O sistema consegue criar diferentes tipos de níveis com pouca participação humana no processo. Esses *grupos de ritmos* são definidos em [70] como períodos alternados de alta e baixa dificuldade. Neste trabalho também foram usadas métricas de *linearidade* e *leniência* para a avaliação do conteúdo final gerado.

Os autores em [71]–[73] usam *padrões* para a construção de fases. Esses *padrões* são pequenas partes do SMB original, os quais vão servir de base estrutural das novas fases criadas. Em dois desses trabalhos, [73] e [72], foram usados algoritmos evolutivos para a geração destes níveis. Nesses trabalhos, também é feita uma avaliação de conteúdo apenas usando as métricas de expressividade definidas em [67].

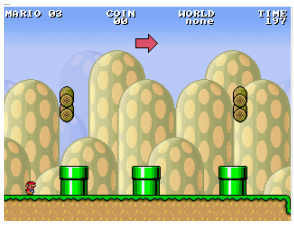
Figura 4 apresenta exemplos de níveis criados para o jogo *Infinite Mario Bros.* com diferentes valores de linearidade, leniência e densidade.



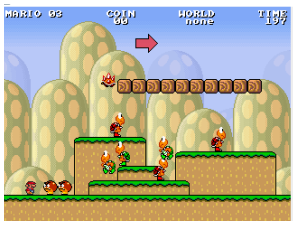
(a) Nível com linearidade alta.



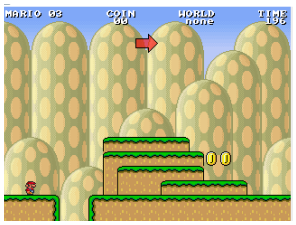
(b) Nível com linearidade baixa.



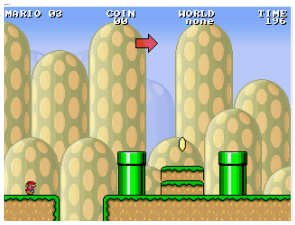
(c) Nível com leniência alta.



(d) Nível com leniência baixa.



(e) Nível com densidade alta.



(f) Nível com densidade baixa.

Fig. 4: Exemplos de níveis do *Infinite Mario Bros.*. O nível da figura (a) tem uma *linearidade* maior que o nível da figura (b) ao apresentar estruturas com alturas similares. O nível da figura (c) tem uma *leniência* maior que o nível figura (d) ao incluir menos inimigos, ficando mais fácil de ser atravessado. O nível da figura (e) tem uma *densidade* maior que o nível da figura (f) ao apresentar maior número de montanhas sobrepostas.

Outros tipos de métricas automáticas menos comuns também são usadas na literatura. Por exemplo, em [74], é proposto um sistema que usa evolução gramatical para geração de níveis de SMB adaptados aos jogadores. Para avaliação, são usados agentes inteligentes com diferentes estilos de jogo para jogarem fases inteiras. A partir de suas sessões de jogo, são atribuídos valores finais ao desempenho adaptativo das fases. A escolha de agentes no lugar de humanos é devida ao grande numero de níveis para serem testados. Para o processo de criação de fases, uma gramática funciona como representação estrutural da fase, e uma função de *fitness* escolhe as melhores fases para o jogador, tentado escolher valores ótimos segundo características particulares dele.

B. AVALIAÇÃO HUMANA

O conteúdo final gerado também pode ser avaliado por seres humanos. Há estudos que incluem a perspectiva humana, normalmente com o jogador atribuindo uma nota à qualidade final do conteúdo gerado. Estudos recentes [75][28] indicam que, apesar das métricas computacionais serem uma forma rápida e barata de avaliação do conteúdo, elas não devem substituir os estudos com seres humanos. Por esse motivo, a criação de métricas que representem de forma mais confiável a perspectiva humana é uma área que ainda demanda estudos.

Um sistema de geração de fases adaptativas para jogos de plataformas é descrito em [76]. Estilos de jogos de cada jogador são aprendidos para criação de novas fases de acordo com a sua forma particular de jogar. Neste trabalho, a opinião dos jogadores é obtida para avaliação dos conteúdos finais. Apesar de utilizarem-se agentes para avaliar os conteúdos finais, os autores usam seres humanos para comparar uma versão adaptativa com uma não adaptativa do mesmo sistema.

Os autores de [77] também propõem a criação de fases personalizadas para o IMB, a partir de fases adaptadas segundo a experiência particular de cada jogador. De forma similar a [76], estudos com seres humanos são feitos para comparar as fases geradas por uma versão adaptativa com uma não adaptativa do sistema proposto.

Nos trabalhos descritos em [24] e [28], focados na geração procedural de níveis para jogos de plataforma, os participantes dos experimentos definem qual, dentre vários sistemas, gera os melhores níveis. Para isso, ao final de cada experimento, os jogadores avaliam o conteúdo em uma escala de 1-7, segundo sua dificuldade, diversão e estética visual. Para determinar qual sistema foi o vencedor, a media das avaliações de todos os usuários é considerada.

V. ÁREAS DE APLICAÇÃO

Desenvolvedores de jogos digitais, voltados ao mercado comercial, utilizam GPC para melhorar seus produtos e acelerar seu desenvolvimento. Por outro lado, pesquisas em GPC são conduzidas utilizando, muitas vezes, jogos existentes no mercado como plataforma de testes. Além do uso de GPC nos chamados jogos comerciais, pesquisadores estão aplicando GPC para os chamados jogos sérios. Trata-se de jogos utilizados para alterar positivamente a realidade, destacando-se aqui aqueles voltados para a reabilitação e educação. Os jogos para reabilitação apresentam conteúdos focados no tratamento de pacientes. Eles podem auxiliar a acelerar a recuperação dos pacientes ou tornar esse processo mais motivador. Os jogos para educação também procuram aumentar a motivação dos estudantes e professores, além de possibilitar uma adequação do aprendizado ao nível de dificuldade do aluno. Esta seção reporta alguns trabalhos considerando o uso da GPC em três categorias de jogos: comerciais, reabilitação e educação.

A. JOGOS COMERCIAIS

Segundo [16] e [21], GPC aplicada a jogos digitais comerciais considera aspectos relevantes como um bom gerenciamento de memória computacional e jogabilidade.

O gerenciamento de memória visa a oferta de bons jogos, apesar das limitações de armazenamento. Esse é o cenário, por exemplo, presente em conteúdos gerados para dispositivos móveis. A jogabilidade foca na oferta de uma experiência diferente a cada novo jogo, onde aspectos subjetivos como altos níveis de entretenimento ou boa estética visual devem ser considerados.

Assim, desenvolvedores aplicam conceitos de GPC para aumentar a diversão de seus jogadores e produzir maior quantidade de conteúdo em um menor período de tempo. Um exemplo de GPC para uso comercial é a IA chamada *The Director*, criada para o jogo *Left 4 Dead* da *Valve* [78]. Essa IA é responsável por decidir onde e quando instanciar inimigos e itens no mapa. Ela toma decisões baseada no comportamento do jogador.

Por exemplo, o jogador pode estar mais relaxando quando seu personagem está fora de uma sequência do jogo com alta intensidade como um combate. Nesse caso, diversos inimigos são instanciados até atingir determinado patamar que leve a uma mudança de comportamento do jogador.

Esse patamar é mantido por algum tempo e, em seguida, a quantidade de inimigos novos é reduzida até o jogador eliminar a ameaça e relaxar novamente. Após certo tempo, a IA retoma o processo de gerar tensão no jogo. Com isso, narrativas únicas são geradas a cada partida de acordo com o comportamento do jogador. O sistema foi bem aceito pelo público e utilizado em diversos outros jogos de sucesso como *Far Cry 3* da *Ubisoft* [79].

Outro jogo que utiliza GPC é *Spore* da *Maxis* [80]. Trata-se de um jogo de simulação de vida, no qual o jogador deve criar e controlar diversas criaturas desde seu estado unicelular até a conquista intergalática. O jogador pode criar e editar seres vivos, adicionando membros e adereços. Também é possível alterar dimensões e formato do corpo e dos adereços, onde um grande número de texturas, cores e padrões são aplicados automaticamente. O processo de animação de cada indivíduo é gerado proceduralmente. Os jogadores podem trocar objetos e criaturas entre si e, graças à GPC usada na geração destes, apenas alguns *kilobytes* são necessários para transmitir cada item. Adicionalmente, o jogo gera música proceduralmente com o *software The Shuffler*, criado especialmente para o jogo. Este programa gera fragmentos de trilha sonora a partir de uma quantia de amostras e foi baseado na linguagem de programação *Pure Data* [81].

Os jogos de tiro em primeira pessoa *Borderlands* da *Gearbox Software*, lançado em 2008, e sua continuação, *Borderlands 2* da mesma desenvolvedora, lançado em 2012 [82], aplicam GPC para a criação de itens e características de inimigos. O jogo conta com mais de 3 milhões de variações de armas que podem alterar seu poder de fogo e precisão; além de adicionar efeitos especiais como queimar o inimigo ou cobri-lo de ácido. Os inimigos gerados pelo sistema apresentam comportamentos diferentes ainda que pertencendo a uma mesma espécie. Por exemplo, um inimigo chamado de "spiderant" pode surgir em duas variantes: a primeira que salta pelo mapa e pula em cima dos jogadores, e a outra que se dobra em uma bola para atacar [83].

Recentemente, o título No Man's Sky da *Hello Games* [84] despertou a atenção da mídia e jogadores por gerar proceduralmente um universo inteiro com cerca de 18 quintilhões de planetas. Cada planeta apresenta flora e fauna únicos, gerados por algoritmos. A música do jogo também utiliza GPC [85]. As criaturas do jogo são criadas através de um programa relativamente simples. Para cada parte do corpo, existe uma lista de elementos com alguns elegíveis e outros não, dependendo de quais elementos já foram selecionados. Logo, um item elegível é selecionado para que um elemento filho seja selecionado em seguida. Por exemplo, para a cabeça de um ser vivo, é selecionado um modelo dentre vários e, em seguida, parte-se para a seleção da boca que está limitada ao conjunto de bocas adequadas ao modelo de cabeça escolhido. Esse modelo é simples computacionalmente (uma travessia em árvore) e pode ser facilmente controlado pelos *designers*. Também há um algoritmo para geração das texturas que se ajusta aos modelos definidos previamente. Mais exemplos de uso de GPC em jogos comerciais podem ser encontrados em [86].

Pesquisas em GPC também utilizam jogos comerciais como plataforma para testar os algoritmos propostos. Os trabalhos apresentados em [62] para *Starcraft*, [60] para *Angry birds* e [24] para *Infinite Mario Bros.* são exemplos de estudos conduzidos usando GPC em jogos comerciais bastante conhecidos.

No exemplo de [60], os pesquisadores criaram um clone do jogo *Angry Birds* como *framework* de testes. Esse clone continha todas as mecânicas e físicas básicas do jogo original, descartando-se apenas a variedade de materiais de blocos, pássaros e porcos. Tal *framework* foi utilizado em outras pesquisas sobre o jogo, como [87] [48] e também foi utilizado como base para a competição de geração de níveis AIBIRDS CIG 2016 [88]. Os autores em [89] abordam o problema de verificação de jogabilidade no jogo *Cut The Rope* [90], onde um agente inteligente é usado para executar simulações completas ou parciais. Uma abordagem de busca em profundidade é aplicada para as simulações, onde restrições são estabelecidas para limitar o espaço de busca.

O problema da geração de nível é abordado através da combinação de GPC e computação humana, ou seja, incluindo a ajuda de seres humanos num processo de geração de níveis semi-automático para o jogo *Infinite Mario Bros.* [28]. O sistema proposto gera níveis inteiros a partir de níveis menores, previamente avaliados por seres humanos. O sistema une esses níveis menores de acordo com uma curva de tensão de dificuldade. Os autores concluem que a participação humana no processo de GPC é necessária para a definição e geração de bons níveis.

Já em [91], os níveis para o mesmo jogo são criados proceduralmente apenas com métricas de *design*, sem intervenção humana. Neste trabalho, um nível é dividido em 4 componentes: solo, blocos, inimigos e moedas. Cada um é evoluído separadamente como diferentes populações

de um AG multipopulacional. O solo é representado por um vetor que contém o valor de sua altura (posição vertical) no nível, enquanto seu índice representa sua posição horizontal. Os outros componentes possuem representação similar, porém, o conteúdo em cada índice de seus vetores indica qual tipo de bloco/moeda/inimigo será instanciado naquela posição, sendo a altura de cada tipo fixada pelos pesquisadores. A avaliação dos indivíduos da população do solo é baseada no conceito de entropia, que mede o grau de imprevisibilidade dos componentes. Já as outras populações são avaliadas pelo conceito de esparsidade, a qual aumenta diretamente com a distância entre dois elementos. O método provou-se eficaz em gerar níveis com características distintas entre si, dados diferentes parâmetros para o AG. Um exemplo de nível gerado pode ser visto na figura 5

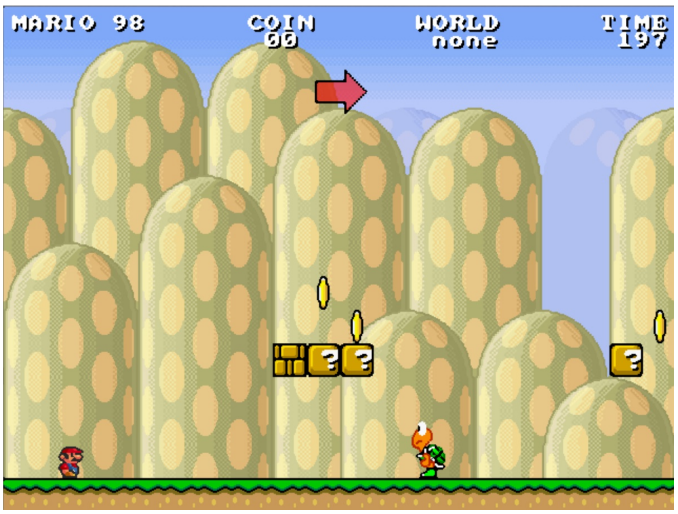


Fig. 5: Nível gerado pelo algoritmo apresentado em [91]. Neste segmento, podemos observar que os vetores de blocos possuiriam valores diferentes de zero nas posições de 8 a 11 e 18 (iniciando-se o índice em 0, representando o primeiro bloco mais à esquerda), enquanto o vetor de inimigos teria apenas um valor diferente de zero, na posição 12. Por fim, o vetor de moedas seria não-nulo nos índices 9, 10 e 19.

B. JOGOS PARA REABILITAÇÃO

Os jogos para reabilitação consideram a geração de conteúdos com regras bem definidas para atender objetivos específicos. Esses objetivos podem ser, por exemplo, a recuperação física ou a reabilitação cognitiva. Nesse contexto, destacam-se vários trabalhos que demonstraram como, para fins de reabilitação, estes jogos podem melhorar e acelerar o processo de terapia ao proporcionar um ambiente mais motivador [92] [93].

Os autores em [94] apresentam um jogo do tipo *esqui-slalom* adaptável para reabilitação de pessoas com pernas lesionadas. Os parâmetros do jogo são ajustados de acordo com o nível de dificuldade ideal para cada jogador, dado seu nível de lesão e suas necessidades personalizadas.

Um motor de jogo de reabilitação adaptativo é descrito em [95]. Através de inteligência computacional, ele permite uma reabilitação mais controlada para o paciente em sua própria casa. O sistema é capaz de dar um retorno online para os pacientes de acordo com o movimento executado. O sistema foi pensado para ser um terapeuta virtual que informa ao paciente o seu progresso. A geração de um nível de dificuldade ideal para cada paciente torna possível um progresso adequado no processo de reabilitação.

De acordo com [96], a reabilitação cognitiva é aquela que foca na correção eficaz de deficiências nas funções cognitivas como memória, atenção, raciocínio e linguagem. Os autores em [97] usam técnicas de IA para o desenvolvimento de um jogo adaptativo orientado ao treinamento de pacientes que sofrem de *alzheimer*. Esses pacientes usualmente perdem faculdades cognitivas como memória, perseverança, planejamento, entre outras. O jogo pode adaptar-se ao jogador segundo o desempenho e ações do mesmo, por meio de técnicas de identificação de erros. Nelas, algoritmos de reconhecimento permitem determinar se uma ação especifica no jogo é considerada como um erro do paciente ou como uma ação dentro de um plano elaborado por ele. Um Jogador com poucos erros poderá receber um conteúdo de jogo mais desafiador, de forma a realizar uma terapia mais adaptada a seu déficit neurológico.

C. JOGOS PARA EDUCAÇÃO

Em outro trabalho [98], um jogo multi-jogador focado na resolução de conflitos é proposto. Nele, tenta-se ensinar a pessoas jovens algumas formas aceitas cultural e socialmente para solucionar conflitos, utilizando métodos pacíficos e construtivos. Alguns dos domínios de conflito considerados no trabalho são salas de aula, eventos esportivos e situações domésticas cotidianas.

Por exemplo, no domínio de uma situação domestica, um objetivo do jogo é a atribuição de tarefas de casa entre vários integrantes de uma família. A dificuldade nesse caso é lidar com a aqueles jogadores que consideram sua tarefa mais complicada que a dos outros. Na pesquisa, são utilizados algoritmos de otimização de partículas e algoritmos evolutivos para a adaptação de diferentes cenários, segundo os domínios de conflito especificados para o projeto.

Um jogo para o ensino de frações matemáticas é apresentado em [99]. O objetivo principal dos autores é criar conteúdo com dificuldade adaptativa, sendo esta adaptação realizada de forma automática de acordo com cada usuário. Para a criação automática de níveis, mapas de conceito hierárquico são utilizados. Neles, nós em diferentes níveis representam habilidades ou conceitos que o jogador deverá aprender antes de passar para um nível superior. Dessa forma, um algoritmo reunirá vários desses nós (conceitos) e o jogador só passara de nível ao demonstrar ter aprendido. A dificuldade é modificada a partir do desempenho do jogador.

Outro jogo para o ensino de matemática é apresentado em [100]. Os autores apresentam um agente inteligente que orienta o aprendizado de fatorização de números, segundo

necessidades particulares do aluno. O agente inteligente ajuda ao estudante quando ele pedir, mas também consegue saber quando e como intervir mesmo sem ser solicitado, segundo um modelo probabilístico do conhecimento do aluno. O modelo está baseado numa rede bayesiana dinâmica que faz o seguimento das ações do aluno e as intervenções do agente durante a sessão de jogo.

VI. CONCLUSÃO

O presente artigo apresentou uma visão geral de trabalhos que empregam GPC em jogos digitais. O principal objetivo é motivar e fornecer alguma orientação aos pesquisadores interessados neste campo de estudo.

Assim, motivações para o uso de GPC nos diversos tipos de conteúdos passíveis de criação por algoritmos foram inicialmente apresentados. Destaca-se que a geração específica de níveis para jogos é um dos tópicos mais abordados pelos pesquisadores. Todavia, a geração de conteúdos vai além deste tópico, possibilitando também criar texturas e modelos, e alterar uma narrativa ou conteúdo musical a partir de eventos relacionados ao comportamento do jogador.

Trabalhos utilizando algoritmos baseados em busca foram revisados, uma vez que tais algoritmos têm sido amplamente empregados na literatura para GPC. Nesse contexto, foram enfatizados o papel da representação da solução e sua relação com o espaço de busca do problema, através da sua codificação e decodificação. Além disso, tal representação deve ser avaliada de forma coerente com os objetivos a serem otimizados no processo de geração de conteúdos. Por isso, as avaliações diretas e simuladas de uma representação foram descritas e os trabalhos relacionados revisados.

Uma vez que o conteúdo é gerado, é necessário avaliar se sua qualidade é satisfatória. Para isso, foram abordados trabalhos que utilizam as duas formas mais comuns de avaliação: métricas computacionais e avaliações humanas. Por último, apresentamos alguns trabalhos considerando a aplicação da GPC em três categorias de jogos: comerciais, reabilitação e educacionais.

A GPC apresenta o potencial para facilitar a criação dos mais diferentes tipos de conteúdos para jogos digitais. Tais conteúdos podem ser criados com qualidade pela GPC, sendo capazes de atender às necessidades dos jogadores, que vão desde o simples lazer até o aprendizado e a reabilitação médica.

Os interessados em se aprofundar no estudo da GPC em jogos devem ler o livro [45] já citado no texto. Para uma abordagem prática, o livro [101], que utiliza a linguagem de programação C++, e o *framework SMFL* [102] são recomendados para aqueles que preferem trabalhar sem o auxílio de motores de jogos. Para os que desejam utilizar *engines*, os autores recomendam o uso do motor *Unity 3D* [103], devido ao seu aprendizado relativamente simples, acompanhado de grande poder computacional. Há extensos e variados tutoriais fornecidos pela própria desenvolvedora do motor e pela comunidade de jogos digitais. Para *Unity 3D*, o livro [104] oferece uma boa introdução ao seu uso para GPC, mas não se aprofunda em técnicas mais complexas.

Para técnicas mais recentes ou mais específicas, o leitor pode buscar dentre os diversos artigos de desenvolvedores de jogos no *blog* sobre desenvolvimento de jogos *Gamasutra* [105]. Também existe uma *Wiki* específica para este assunto [106], com diversos conceitos básicos e avançados, além de referências para maiores detalhes de cada assunto. Por último, o fórum *Reddit* possui um *subreddit* (uma espécie de fórum específico para o assunto) em GPC no qual muitos programadores dividem suas experiências e disponibilizam tutoriais de diferentes níveis de dificuldade com os mais diversos algoritmos [107].

Referências

[1] J. Huizinga, *Homo ludens*. Perspectiva, 2012.

[2] Newzoo, *The global games market reaches $99.6 billion in 2016, mobile generating 37%*. [Online]. Available: https://newzoo.com/insights/articles/global-games-market-reaches-99-6-billion-2016-mobile-generating-37/.

[3] ——, *Newzoo summer series #13: Brazilian games market*. [Online]. Available: https://newzoo.com/insights/infographics/newzoo-summer-series-13-brazilian-games-market/.

[4] ——, *A regional breakdown of the $99.6 bn global games market (free report)*. [Online]. Available: https://newzoo.com/insights/markets/games/.

[5] V. Corporation, *Steam, a plataforma definitiva de entretenimento*. [Online]. Available: http://store.steampowered.com/about/.

[6] E. E. Design and R. (EEDAR), *Awesome video game data*. [Online]. Available: http://www.eedar.com/Pres/EEDAR%20-%20GDC2016%20-Awesome%20Video%20Game%20Data%20Distribute%20%5BGeoffrey%20Zatkin%5D%20v2.7.pdf.

[7] A. Inc., *Itunes - apple*. [Online]. Available: https://www.apple.com/itunes/.

[8] S. .-.-. T. S. Portal, *Number of newly developed applications/games submitted for release to the itunes app store from 2012 to 2016*. [Online]. Available: https://www.statista.com/statistics/258160/number-of-new-apps-submitted-to-the-itunes-store-per-month/.

[9] E. S. A. (ESA), *2015 annual report*. [Online]. Available: http://www.theesa.com/wp-content/uploads/2016/04/ESA-Annual-Report-2015-1.pdf.

[10] A. Iosup, "Poggi: Generating puzzle instances for online games on grid infrastructures," 2, vol. 23, Chichester, UK: John Wiley and Sons Ltd., Feb. 2011, pp. 158–171. DOI: 10.1002/cpe.1638. [Online]. Available: http://dx.doi.org/10.1002/cpe.1638.

[11] R. Weber, *On reflections: First interview with the ubisoft studio's new md*. [Online]. Available: http://www.gamesindustry.biz/articles/2014-02-26-on-reflections-first-interview-with-the-ubisoft-studios-new-md.

[12] J. Brightman, *Warren spector: A lifetime of achievements*. [Online]. Available: http://www.gamesindustry.biz/articles/2012-03-16-warren-spector-a-lifetime-of-achievements.

[13] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," 1, vol. 9, New York, NY, USA: ACM, Feb. 2013, 1:1–1:22. DOI: 10.1145/2422956.2422957. [Online]. Available: http://doi.acm.org/10.1145/2422956.2422957.

[14] B. Fritz and A. Pham, *Star wars: The old republic — the story behind a galactic gamble*. [Online]. Available: http://herocomplex.latimes.com/games/star-wars-the-old-republic-the-story-behind-a-galactic-gamble/#/0.

[15] J. Togelius, E. Kastbjerg, D. Schedl, and G. N. Yannakakis, "What is procedural content generation?: Mario on the borderline," in *Proceedings of the 2Nd International Workshop on Procedural Content Generation in Games*, ser. PCGames '11, Bordeaux, France: ACM, 2011, 3:1–3:6, ISBN: 978-1-4503-0872-4. DOI: 10.1145/2000919.2000922. [Online]. Available: http://doi.acm.org/10.1145/2000919.2000922.

[16] G. Smith, E. Gan, A. Othenin-Girard, and J. Whitehead, "Pcg-based game design: Enabling new play experiences through procedural content generation," in *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*, ACM, 2011, p. 7.

[17] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 9, no. 1, p. 1, 2013.

[18] R. Khaled, M. J. Nelson, and P. Barr, "Design metaphors for procedural content generation in games," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '13, Paris, France: ACM, 2013, pp. 1509–1518, ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2466201. [Online]. Available: http://doi.acm.org/10.1145/2470654.2466201.

[19] I. Bell and D. Braben. (1984). The elite home page, [Online]. Available: http://www.iancgbell.clara.net/elite/.

[20] Mossmouth. (2013). Spelunky, [Online]. Available: http://www.spelunkyworld.com/.

[21] N. Shaker, J. Togelius, and M. J. Nelson, *Procedural content generation in games: A textbook and an overview of current research*. Springer, 2016.

[22] E. Byrne, *Game level design*. Charles River Media Boston, 2005.

[23] A. Khalifa, D. Perez-Liebana, S. M. Lucas, and J. Togelius, "General video game level generation,"

[24] J. R. Mariño and L. H. Lelis, "A computational model based on symmetry for generating visually pleasing maps of platform games," in *Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2016.

[25] M. Preuss, A. Liapis, and J. Togelius, "Searching for good and diverse game levels," in *2014 IEEE Conference on Computational Intelligence and Games*, IEEE, 2014, pp. 1–8.

[26] T. Iwatani, "Pac-man," *Tokyo: Namco*, 1980.

[27] R. J. V. de Medeiros and T. F. V. de Medeiros, "Procedural level balancing in runner games," in *2014 Brazilian Symposium on Computer Games and Digital Entertainment*, IEEE, 2014, pp. 109–114.

[28] W. M. Reis, L. H. Lelis, *et al.*, "Human computation for procedural content generation in platform games," in *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, 2015, pp. 99–106.

[29] B. S. LLC. (2016). Doom official website, [Online]. Available: http://doom.com/en-us/.

[30] E. Booth, J. Thangarajah, and F. Zambetta, "Flexible story generation with norms and preferences in computer role playing games," in *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, 2015, pp. 68–74.

[31] S. P. Hernandez, V. Bulitko, and E. S. Hilaire, "Emotion-based interactive storytelling with artificial intelligence.," in *AIIDE*, 2014.

[32] S. P. Hernandez, V. Bulitko, and M. Spetch, "Keeping the player on an emotional trajectory in interactive storytelling," in *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*, 2015.

[33] A. Ramirez and V. Bulitko, "Automated planning and player modeling for interactive storytelling," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 7, no. 4, pp. 375–386, 2015.

[34] G. Lee, V. Bulitko, and E. A. Ludvig, "Sports commentary recommendation system (scores): Machine learning for automated narrative.," in *AIIDE*, 2012.

[35] K. Collins, "An introduction to procedural music in video games," *Contemporary Music Review*, vol. 28, no. 1, pp. 5–15, 2009.

[36] D. Plans and D. Morelli, "Experience-driven procedural music generation for games," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 3, pp. 192–198, 2012.

[37] M. Eladhari, R. Nieuwdorp, and M. Fridenfalk, "The soundtrack of your mind: Mind music-adaptive audio for game characters," in *Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, ACM, 2006, p. 54.

[38] W. Cachia, L. Aquilina, H. P. Martinez, and G. N. Yannakakis, "Procedural generation of music-guided weapons," in *2014 IEEE Conference on Computational Intelligence and Games*, IEEE, 2014, pp. 1–2.

[39] A. Jordan, D. Scheftelowitsch, J. Lahni, J. Hartwecker, M. Kuchem, M. Walter-Huber, N. Vortmeier, T. Delbrügger, Ü. Güler, I. Vatolkin, *et al.*, "Beatthebeat music-based procedural content generation in a mobile game," in *2012 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, 2012, pp. 320–327.

[40] Y. Manavalan, V. Bulitko, and M. Spetch, "A lightweight algorithm for procedural generation of emotionally affected behavior and appearance," in *Eleventh Artificial Intelligence and Interactive Digital Entertainment Conference*, 2015.

[41] G. d. O. B. Leite and E. S. de Lima, "Geração procedural de mapas para jogos 2d,"

[42] A. Liapis, G. N. Yannakakis, and J. Togelius, "Sentient sketchbook: Computer-aided game level authoring.," in *FDG*, 2013, pp. 213–220.

[43] J. Togelius, G. Yannakakis, K. Stanley, and C. Browne, "Search-based procedural content generation: A taxonomy and survey," *Computational Intelligence and AI in Games, IEEE Transactions on*, vol. 3, no. 3, pp. 172–186, Sep. 2011. DOI: 10.1109/TCIAIG.2011.2148116.

[44] M. Gendreau and J.-Y. Potvin, *Handbook of metaheuristics*, 2nd. Springer Publishing Company, Incorporated, 2010, ISBN: 1441916636, 9781441916631.

[45] N. Shaker, J. Togelius, and M. J. Nelson, Eds., *Procedural content generation in games: A textbook and an overview of current research*. pcgbook.com, 2013.

[46] S. Dahlskog and J. Togelius, "Patterns and procedural content generation: Revisiting Mario in world 1 level 1," in *ACM International Conference Proceeding Series*, 2012.

[47] P. Bentley and S. Kumar, "Three ways to grow designs: A comparison of evolved embryogenies for a design problem," in *Genetic and Evolutionary Computation Conference*, Morgan Kaufmann, 1999, pp. 35–43.

[48] L. T. Pereira, C. Toledo, L. N. Ferreira, and L. H. Lelis, "Learning to speed up evolutionary content generation in physics-based puzzle games,"

[49] Rovio. (2009). Angry birds, [Online]. Available: https://www.angrybirds.com/.

[50] N. Shaker, M. Shaker, and J. Togelius, "Evolving playable content for cut the rope through a simulation-based approach.," in *AIIDE*, G. Sukthankar and I. Horswill, Eds., AAAI, 2013. [Online]. Available: http://dblp.uni-trier.de/db/conf/aiide/aiide2013.html#ShakerST13.

[51] L. Cardamone, D. Loiacono, and P. L. Lanzi, "Interactive evolution for the procedural generation of tracks in a high-end racing game," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '11, Dublin, Ireland: ACM, 2011, pp. 395–402, ISBN: 978-1-4503-0557-0. DOI: 10.1145/2001576.2001631.

[52] D. Loiacono, L. Cardamone, and P. Lanzi, "Automatic track generation for high-end racing games using evolutionary computation," *Computational Intelligence and AI in Games, IEEE Transactions on*, vol. 3, no. 3, pp. 245–259, Sep. 2011. DOI: 10.1109/TCIAIG.2011.2163692.

[53] S. O. Kimbrough, G. J. Koehler, M. Lu, and D. H. Wood, "On a feasible-infeasible two-population (fi-2pop) genetic algorithm for constrained optimization: Distance tracing and no free lunch.," *European Journal of Operational Research*, vol. 190, no. 2, pp. 310–327, 2008.

[54] A. Liapis, G. N. Yannakakis, and J. Togelius, "Enhancements to constrained novelty search: Two-population novelty search for generating game content," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '13, Amsterdam, The Netherlands: ACM, 2013, pp. 343–350, ISBN: 978-1-4503-1963-8. DOI: 10.1145/2463372.2463416. [Online]. Available: http://doi.acm.org/10.1145/2463372.2463416.

[55] G. N. Yannakakis and J. Togelius, "Experience-Driven Procedural Content Generation," vol. 2, no. 3, pp. 147–161, 2011. DOI: 10.1109/T-AFFC.2011.6.

[56] M. Cook and S. Colton, "Multi-faceted evolution of simple arcade games," in *Computational Intelligence and Games (CIG), 2011 IEEE Conference on*, Aug. 2011, pp. 289–296. DOI: 10.1109/CIG.2011.6032019.

[57] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: A mixed-initiative level design tool," in *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, ser. FDG '10, Monterey, California: ACM, 2010, pp. 209–216, ISBN: 978-1-60558-937-4. DOI: 10.1145/1822348.1822376. [Online]. Available: http://doi.acm.org/10.1145/1822348.1822376.

[58] J. Bird, T. Feltwell, and G. Cielniak, "Real-time adaptive track generation in racing games," in *13th International Conference on Intelligent Games and Simulation, GAME-ON 2012*, School of Computer Science, University of Lincoln, United Kingdom, 2012, pp. 48–52.

[59] N. Shaker, M. Shaker, and J. Togelius, "Ropossum: An authoring tool for designing, optimizing and solving cut the rope levels.," in *AIIDE*, G. Sukthankar and I. Horswill, Eds., AAAI, 2013. [Online]. Available: http://dblp.uni-trier.de/db/conf/aiide/aiide2013.html#ShakerST13a.

[60] L. Ferreira and C. F. M. Toledo, "A search-based approach for generating angry birds levels," in *Computational Intelligence and Games (CIG), 2014 IEEE Conference on*, IEEE, 2014, pp. 1–8.

[61] W. L. Raffe, F. Zambetta, X. Li, and K. O. Stanley, "Integrated approach to personalized procedural map generation using evolutionary algorithms,"

IEEE Transactions on Computational Intelligence and AI in Games, vol. 7, no. 2, pp. 139–155, 2015.

[62] J. Togelius, M. Preuss, N. Beume, S. Wessing, J. Hagelbäck, and G. N. Yannakakis, "Multiobjective exploration of the starcraft map space," in *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*, IEEE, 2010, pp. 265–272.

[63] B. Entertainment. (1998). Starcraft, [Online]. Available: http://us.blizzard.com/pt-br/games/sc/.

[64] A. S. Ruela and F. G. Guimaraes, "Coevolutionary procedural generation of battle formations in massively multiplayer online strategy games," in *2014 Brazilian Symposium on Computer Games and Digital Entertainment*, IEEE, 2014, pp. 89–98.

[65] M. Kaidan, T. Harada, C. Y. Chu, and R. Thawonmas, "Procedural generation of angry birds levels with adjustable difficulty," in *Evolutionary Computation (CEC), 2016 IEEE Congress on*, IEEE, 2016, pp. 1311–1316.

[66] P. Zhang and J. Renz, *Qualitative spatial representation and reasoning in angry birds: The extended rectangle algebra*, 2014.

[67] G. Smith and J. Whitehead, "Analyzing the expressive range of a level generator," in *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, ACM, 2010, p. 4.

[68] B. Horn, S. Dahlskog, N. Shaker, G. Smith, and J. Togelius, "A comparative evaluation of level generators in the Mario AI framework.," in *FDG*, ACM, 2014.

[69] G. Smith, M. Treanor, J. Whitehead, M. Mateas, M. Treanor, J. March, and M. Cha, "Launchpad: A rhythm-based level generation for 2d platformers," *IEEE Transactions on Computing Intelligence and AI in Games*, vol. 3, no. 1, pp. 1–16, 2011.

[70] G. Smith, M. Cha, and J. Whitehead, "A framework for analysis of 2d platformer levels," in *ACM SIGGRAPH Symposium on Video Games*, ACM, 2008, pp. 75–80.

[71] S. Dahlskog, J. Togelius, and M. J. Nelson, "Linear levels through n-grams," in *Proceedings of the International Academic MindTrek Conference*, 2014.

[72] S. Dahlskog and J. Togelius, "A multi-level level generator," in *IEEE Conference on Computational Intelligence and Games*, 2014, pp. 1–8.

[73] ——, "Procedural content generation using patterns as objectives," in *Proceedings of the European Conference Applications of Evolutionary Computation*, 2014, pp. 325–336.

[74] N. Shaker, G. N. Yannakakis, J. Togelius, M. Nicolau, and M. O'Neill, "Evolving personalized content for Super Mario Bros using grammatical evolution," in *Conference on Artificial Intelligence and Interactive Digital Entertainment*, AAAI Press, 2012, pp. 75–80.

[75] J. Mariño, W. Reis, and L. Lelis, "An empirical evaluation of evaluation metrics of procedurally generated mario levels," *AIIDE*, pp. 44–50, 2015.

[76] N. Shaker, G. N. Yannakakis, and J. Togelius, "Towards automatic personalized content generation for platform games," in *Conference on Artificial Intelligence and Interactive Digital Entertainment*, AAAI Press, 2010, pp. 63–68.

[77] S. Bakkes, S. Whiteson, G. Li, G. V. Visniuc, E. Charitos, N. Heijne, and A. Swellengrebel, "Challenge balancing for personalised game spaces," in *Games Media Entertainment*, IEEE Press, 2014, pp. 1–8.

[78] M. Booth, *The ai systems of left 4 dead*. [Online]. Available: http://www.valvesoftware.com/publications/2009/ai_systems_of_l4d_mike_booth.pdf.

[79] Ubisoft. (2012). Far cry 3, [Online]. Available: https://www.ubisoft.com/pt-BR/game/far-cry-3/.

[80] Maxis. (2008). Spore, [Online]. Available: http://www.spore.com/.

[81] Wikipedia. (2008). Development of spore, [Online]. Available: https://en.wikipedia.org/wiki/Development_of_Spore.

[82] G. Software. (2008). Borderlands - home page, [Online]. Available: https://borderlandsthegame.com/.

[83] Wikipedia. (2008). Borderlands(video game), [Online]. Available: https://en.wikipedia.org/wiki/Borderlands_(video_game%29.

[84] H. Games, *No man's sky*. [Online]. Available: http://www.no-mans-sky.com/about/.

[85] 3. G. D. Blog, *No man's sky – procedural content*. [Online]. Available: http://3dgamedevblog.com/wordpress/?p=836.

[86] I. Lambe, *Procedural content generation: Thinking with modules*. [Online]. Available: http://www.gamasutra.com/view/feature/174311/procedural_content_generation_.php.

[87] L. Ferreira and C. Toledo, "Generating levels for physics-based puzzle games with estimation of distribution algorithms," in *Proceedings of the 11th International Conference on Advances in Computer Entertainment*, ser. ACE'14, Funchal, Portugal, 2014. DOI: 10.1145/2663806.2663847. [Online]. Available: http://www.lucasnferreira.com/papers/2014/ace-edaab.pdf.

[88] A. Birds.org. (2016). Aibirds cig 2016 level generation competition, [Online]. Available: https://aibirds.org/other-events/level-generation-competition.html.

[89] N. Shaker, M. Shaker, and J. Togelius, "Evolving playable content for cut the rope through a simulation-based approach.," in *AIIDE*, 2013.

[90] ZeptoLab. (2010). Cut the rope, [Online]. Available: https://www.zeptolab.com/games/cut_the_rope.

[91] L. Ferreira, L. Pereira, and C. Toledo, "A multi-population genetic algorithm for procedural generation of levels for platform games," in *Proceedings of the 16th Conference on Genetic and Evolutionary Computation*, ser. GECCO'14, Vancouver, Canada, 2014, p. 45. DOI: 10.1145/2598394.2598489. [Online]. Available: http://www.lucasnferreira.com/papers/2014/gecco-mario.pdf.

[92] K. J. Bower, J. Louie, Y. Landesrocha, P. Seedy, A. Gorelik, and J. Bernhardt, "Clinical feasibility of interactive motion-controlled games for stroke rehabilitation," *Journal of neuroengineering and rehabilitation*, vol. 12, no. 1, p. 1, 2015.

[93] A. Shapi'i, N. N. Bahari, H. Arshad, N. A. M. Zin, and Z. R. Mahayuddin, "Rehabilitation exercise game model for post-stroke using microsoft kinect camera," in *Biomedical Engineering (ICoBE), 2015 2nd International Conference on*, IEEE, 2015, pp. 1–6.

[94] D. Dimovska, P. Jarnfelt, S. Selvig, and G. N. Yannakakis, "Towards procedural level generation for rehabilitation," in *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*, ACM, 2010, p. 7.

[95] M. Pirovano, R. Mainetti, G. Baud-Bovy, P. L. Lanzi, and N. A. Borghese, "Intelligent game engine for rehabilitation (iger)," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 8, no. 1, pp. 43–55, 2016.

[96] M. M. Sohlberg and C. A. Mateer, *Introduction to cognitive rehabilitation: Theory and practice*. Guilford Press, 1989.

[97] F. Imbeault, B. Bouchard, and A. Bouzouane, "Serious games in cognitive training for alzheimer's patients," in *Serious Games and Applications for Health (SeGAH), 2011 IEEE 1st International Conference on*, IEEE, 2011, pp. 1–8.

[98] G. N. Yannakakis, J. Togelius, R. Khaled, A. Jhala, K. Karpouzis, A. Paiva, and A. Vasalou, "Siren: Towards adaptive serious games for teaching conflict resolution,"

[99] E. Andersen, "Optimizing adaptivity in educational games," in *Proceedings of the International Conference on the Foundations of Digital Games*, ACM, 2012, pp. 279–281.

[100] C. Conati and X. Zhao, "Building and evaluating an intelligent pedagogical agent to improve the effectiveness of an educational game," in *Proceedings of the 9th international conference on Intelligent user interfaces*, ACM, 2004, pp. 6–13.

[101] D. Green, *Procedural content generation for c++ game development*. Packt Publishing Ltd., 2016.

[102] L. Gomila, *Simple and fast multimedia library (sfml)*. [Online]. Available: http://www.sfml-dev.org/.

[103] U. Technologies, *Unity 3d*. [Online]. Available: https://unity3d.com/pt.

[104] R. Watkins, *Procedural content generation for unity game development: Harness the power of procedural content generation to design unique games with unity*. Packt Pub, 2016.

[105] UBM, *Gamasutra - the art & business of making games*. [Online]. Available: http://www.gamasutra.com/.

[106] Wikidot.com, *Procedural content generation wiki*. [Online]. Available: http://pcg.wikidot.com/.

[107] reddit inc., *Proceduralgeneration*. [Online]. Available: https://www.reddit.com/r/proceduralgeneration/.

Julian Ricardo Hernandez Mariño Possui graduação em Engenharia de Sistemas pela Universidad Industrial de Santander (2012) e mestrado em Ciência da Computação pela Universidade Federal de Viçosa (2016). Atualmente realiza doutorado em Ciência da Computação e Matemática Computacional pela Universidade de São Paulo.

Leonardo Tórtoro Pereira Possui graduação em Ciências de Computação pela Universidade de São Paulo (2016) e atualmente cursa mestrado em Ciência de Computação e Matemática Computacional, também na Universidade de São Paulo.

Claudio Fabiano Motta Toledo Possui graduação em Matemática Aplicada e Computacional pela Universidade Estadual de Campinas (1995), mestrado em Engenharia Elétrica pela Universidade Estadual de Campinas (1999) e doutorado em Engenharia Elétrica pela Universidade Estadual de Campinas (2005). Realizou pós-doutorado no Computer Science and Artificial Intelligence Laboratory do Massachusetts Institute of Technology (2015). Atualmente é professor doutor na Universidade de São Paulo. Tem experiência na área de Ciência da Computação com ênfase em Sistemas Evolutivos e Modelagem Matemática aplicados a problemas de planejamento de produção, otimização, veículos aéreos não tripulados, jogos digitais, recuperação de imagens, entre outros.