

Gerando Cenários Procedurais Adaptáveis em Um Jogo Mobile

Kévin C. de Sá
Márcio G. Martins

Matheus H. Wichman
Cristiano A. da Costa

Universidade do Vale do Rio dos Sinos – Unisinos

Programa de Pós-Graduação em Computação Aplicada – PIPCA, Brasil



Figura 1: Imagens do jogo que utiliza a geração procedural de conteúdo proposta neste artigo.

Abstract

This paper's goal is to describe the implementation of a procedural content generation (PCG) method for adaptable sceneries, corresponding to the player's abilities. This method was applied in a game of the Endless Runner genre, for the iOS platform. This resulted in a procedural game that adjusts itself according to the player's skills in-game, having both simple/easy game levels for beginners and difficult/complex game levels for the more experienced ones. Finally, we conducted an experiment that consisted on users playing and testing the game for a few times, then they had to answer a quick survey, adopting the Likert concept, about the game's acceptance overall and the efficiency of the procedural content generation method proposed on this paper. The results showed that the method succeeded on its objectives, which were to generate balanced and diverse scenario segments that adjusts themselves to the player's skill level.

Palavras-chave: geração procedural, mobile, games.

Informações para Contato:

[\[kvdesa, matheus.wichman\]@gmail.com](mailto:[kvdesa, matheus.wichman]@gmail.com)
[\[cac, marciog\]@unisinos.br](mailto:[cac, marciog]@unisinos.br)

1. Introdução

Jogos do gênero Endless (ou Infinite) Runner são jogos de plataforma onde o personagem está constantemente avançando por um mundo sem fim e, geralmente, gerado de forma procedural [Medeiros e Medeiros 2014]. Os controles e ações do usuário são simples, implicam o jogador a pular e atacar; o principal objetivo é percorrer a maior distância possível sem perder a partida. Devido à simplicidade dos controles, jogos que pertencem à esse gênero se adaptam de forma harmônica e precisa em plataformas móveis (celulares e tablets), onde cada ação efetuada pelo jogador é interpretada como um toque ou deslize do dedo na tela do dispositivo.

Pelo fato de jogos desse estilo possuírem cenários infinitos, a ideia de utilizar geração procedural de conteúdo (GPC) é comum. Por isso, foi empregado GPC para gerar cenários sem fim, mas que também sejam balanceados em dificuldade para cada jogador dependendo de suas habilidades no jogo. Segundo o conceito de fluxo de Csikszentmihaly [Csikszentmihaly 1991], o jogador se encontra em estado de fluxo se ele, ao executar alguma tarefa, estiver totalmente concentrado nesta de forma a perder o senso de si próprio. Para alcançar esse estado, o desafio proposto pelas tarefas deve ser balanceado. Através de GPC, é possível proporcionar tarefas/desafios aos jogadores e induzi-los à esse estado de fluxo, desde que tais desafios sejam

balanceados, esse é um dos pontos tratados neste artigo.

O jogo no qual essa técnica foi empregada, *Dashing Aliens*, foi desenvolvido para a plataforma iOS [Apple 2016] (iPhones e iPads) através do ambiente de desenvolvimento da Apple, o Xcode [Apple Developer 2016], ferramenta utilizada tanto para a criação de softwares para os computadores Mac, que possuem como sistema operacional o OS X [Apple 2016], quanto para os dispositivos iOS. Foi utilizada a linguagem de programação Objective-C [Apple Developer 2016], uma das linguagens de programação utilizadas para o desenvolvimento de softwares para tais plataformas alvo; outra linguagem disponível, e mais recente, é a linguagem de programação Swift [Apple 2016], apresentada no ano de 2014 pela Apple e tendo sua última versão estável lançada em maio de 2016. O jogo foi produzido com o auxílio da Cocos2D [Cocos2D 2016], um framework de software de código livre usado para a construção de jogos, aplicativos e programas interativos.

Na seção seguinte é realizada uma descrição do conceito de GPC. Na seção 3 são denotados os trabalhos relacionados. Na seção 4 é apresentada uma breve descrição do jogo que utiliza o método de GPC desenvolvido neste trabalho. Na quinta seção é denotado o método de geração de cenários procedurais adaptáveis desenvolvido. Na seção 6 é descrito o método de avaliação, bem como os resultados obtidos. Finalmente, na sétima seção são apresentadas as conclusões.

2. Geração Procedural de Conteúdo

“Geração procedural de conteúdo (GPC) em jogos digitais refere-se à criação automática ou semiautomática de conteúdo usando algoritmos” [Dahlskog and Togelius 2012]. Uma das mais importantes, se não a principal, finalidades de GPC em jogos é a de promover um jogo que seja possível jogar diversas vezes sem que essa experiência se torne entediante. Outro conceito relacionado a GPC é a adaptabilidade: a possibilidade do jogo se adaptar conforme as ações e habilidades do jogador, de forma que este se adapte ao estilo de jogar de cada usuário, ampliando a quantidade de possíveis jogadores [Smith et al. 2011].

Esses conceitos são extremamente importantes para jogos do estilo *Infinite Runner*, visto que são títulos que possuem somente uma fase disponível ao jogador. Caso não fosse utilizado algum tipo de GPC para a geração dessa fase, o jogo rapidamente se tornaria repetitivo, pois a cada nova partida o jogador teria de enfrentar a mesma fase, constituída pelos mesmo obstáculos e power ups, em suas mesmas posições das partidas anteriores. Dessa forma, após uma única fase, o usuário já teria o conhecimento de tudo que o jogo tem a oferecer, resultando em um título que não possui apelo com seus jogadores.

3. Trabalhos Relacionados

Nesta seção serão brevemente explicadas as diferentes técnicas de geração procedural de conteúdo que foram pesquisadas a fim de definir o método de geração de cenários procedurais adaptáveis empregado no jogo *Dashing Aliens*. Tais técnicas são descritas nos trabalhos de Polymorph [Jennings-Teats et al. 2010] e Tanagra [Smith et al. 2011].

3.1 Polymorph

Polymorph [Jennings-Teats et al. 2010] é um jogo 2D de plataforma que emprega técnicas que buscam resolver um problema muito frequente em jogos: a disparidade nas habilidades e na aprendizagem de todos os jogadores em relação ao jogo. Para isso, utiliza-se métodos de geração de fases e aprendizagem de máquina para determinar o nível de dificuldade do cenário tanto quanto o nível de habilidade de cada player, construindo e ajustando, dinamicamente, as fases e garantindo que o desafio proposto pela mesma ao usuário seja apropriado.

Os autores de Polymorph desenvolveram uma ferramenta de coleta de dados que exige ao usuário jogar um segmento de fase (aproximadamente 10 segundos de duração), reunindo informações do comportamento e das ações do jogador, tais como: a quantidade de tempo que ele permanece parado ou se movendo para trás, o tempo total que ele demorou para terminar o segmento de fase, a quantidade de moedas coletadas, se o jogador morreu ou completou essa fração do cenário, entre outras. Após o término do segmento (através do sucesso ou da derrota), o usuário deve classificar, entre 1 e 6, esse pedaço de fase conforme a sua dificuldade, sendo 1-Fácil e 6-Difícil. Utilizando pequenas porções de cenário é possível determinar qual elemento está resultando na dificuldade determinada pelo jogador. Após essa etapa de coleta de dados, é feito um ranking dos segmentos de fase conforme a sua dificuldade.

Enquanto o usuário está jogando, Polymorph está sempre avaliando o comportamento do jogador. Cada novo segmento de fase escolhido depende do desempenho atual do usuário, se ele estiver com uma performance baixa, será escolhida uma porção de fase com uma dificuldade baixa e vice-versa. Dessa forma, conforme o jogador aperfeiçoa suas habilidades, a dificuldade é agravada, resultando em um desafio apropriado para cada usuário. Alternadamente, caso o jogador esteja realizando um grande esforço para ter sucesso na fase, a dificuldade da mesma é reduzida.

3.2 Tanagra

Tanagra [Smith et al. 2011], atualmente em sua versão de protótipo, é uma ferramenta de design de cenários para jogos de plataforma 2D. Essa ferramenta auxilia o designer (humano) no processo de criação e produção de diferentes cenários. O designer pode tanto posicionar diferentes geometrias no gerador de fases

quanto determinar o pacing do cenário através da modificação da timeline dos beats; após o término dessa etapa, a ferramenta é responsável por preencher o restante da fase, garantindo que a mesma seja jogável.

Tanagra representa a fase como um conjunto de beats, cada um constituído por componentes de cenário, também chamados de geometrias; as geometrias suportadas são: plataformas, buracos, molas, inimigos, e esmagadores (objeto fixo no eixo X que está sempre realizando um movimento vertical de cima para baixo e possui como objetivo esmagar o jogador). Cada beat encapsula uma única ação do jogador, como pular, que pode ocorrer a qualquer momento desde o início e o fim do beat; as propriedades existentes de um beat são: tempo de início, tempo de fim, um conjunto de geometrias, e conhecimento dos beats vizinhos. O designer pode adicionar, remover ou modificar um beat a qualquer momento e, caso o faça, Tanagra irá propagar as modificações feitas para todo o cenário.

Essa ferramenta gera sugestões de cenários para o designer, garante que todas as fases criadas sejam jogáveis e suporta a edição em tempo real, sendo possível a edição tanto em questão das geometrias e componentes de cenário quanto a adição/remoção/modificação dos beats. Quando Tanagra é inicializada, o designer deve especificar o comprimento da timeline (conjunto de beats) e a quantidade de beats contidas nela; então é apresentada ao usuário a tela de geometrias vazia. Um painel de controle no lado direito da ferramenta permite: adicionar ou remover beats, modificar o comprimento de um beat, selecionar geometria para ser fixada em uma posição específica ou para ser movida na tela, e solicitar um cenário novo que respeite todas as mudanças realizadas pelo designer.

4. O Jogo

É um jogo singleplayer do estilo Endless Runner onde o jogador controla o personagem principal e possui como objetivo percorrer a maior distância possível desviando de obstáculos; quando o usuário bater o seu recorde, o mesmo é salvo e, ao término das partidas, é demonstrado para incentivar a melhoria de desempenho do jogador. Existem diferentes tipos de power ups, que servem para auxiliar o jogador durante a fase, e obstáculos, que servem como empecilho para o jogador.

No início da partida o personagem nasce em uma posição em X fixa. Caso ele colida com algum obstáculo ou fique preso entre plataformas, ele retrocede uma distância determinada. Existem duas formas de uma partida ser encerrada: i. o jogador sair da tela (cair em vãos entre plataformas, por exemplo) ou ii. colidir com um tipo de obstáculo específico, o perseguidor. O canto esquerdo da tela está preenchido por esse último tipo de obstáculo que são serras que perseguem o jogador. Quando o jogador morre, a fase é imediatamente encerrada e é apresentada ao jogador uma tela que indica a distância percorrida pelo mesmo

e o seu recorde atual (caso o usuário tenha superado seu recorde nessa partida, essa informação também é apresentada).

A movimentação do personagem ocorre através de um sistema de impulsos, pulos e paradas: o jogador sempre se encontra na extremidade esquerda da tela, com sua posição inicial no eixo X fixa, podendo variar conforme for atingido por obstáculos ou ativar certos power ups. O usuário pode efetuar o tradicional pulo, para isso ele deve tocar na tela, dessa forma o personagem realiza um simples pulo no eixo Y; também é possível carregar esse pulo, quanto mais tempo segurar o dedo na tela, maior será o pulo. Lembrando que, para pular, o personagem deve estar no chão, não havendo possibilidade de pular no ar.

Quando estiver no ar, o usuário pode realizar impulsos ou paradas. Paradas são ativadas com um simples toque na tela e, quando ativadas, desfazem todo e qualquer movimento vertical do jogador, fazendo com que ele flutue no ar por um instante. Para os impulsos, também no ar, o usuário deve pressionar o dedo na tela, dessa forma o personagem carrega o impulso. Quando o dedo for solto da tela, o personagem irá se lançar na direção do toque com uma determinada velocidade: quanto maior for o tempo que o jogador manter a tela do dispositivo pressionada, maior será a velocidade do lançamento.

Vale ressaltar que o impulso faz com que o personagem se movimente com uma velocidade maior que a tradicional no eixo X e, com o tempo, essa velocidade se estabiliza para seu estado normal; enquanto que a realização de um pulo ou parada não acarreta no acréscimo da velocidade. Para dar uma noção de velocidade e de movimento do personagem (que possui sua posição no eixo X fixa) o mundo está em constante movimento horizontal, da direita para a esquerda. Essas mecânicas de movimento devem ser utilizadas para se desviar dos obstáculos e avançar no cenário.

Existem diferentes elementos com o qual o jogador pode interagir no jogo podendo, ou não, auxiliá-lo durante a partida. As categorias são: power ups, moedas e obstáculos.

Tipo	Descrição
Invencibilidade	Faz com que a colisão do jogador com os obstáculos seja ignorada.
Câmera Lenta	Diminui a velocidade de todos os elementos do jogo.
Vida Extra	Move o jogador para frente uma determinada distância.

Tabela 1: Power ups existentes no jogo

Os power ups auxiliam o jogador e possuem um tempo determinado de vida após serem ativados, ou seja, quando o tempo esgotar, o poder é desativado.

Veja a Tabela 1 para uma explicação detalhada de cada um dos power ups.

Tipo	Descrição
Perseguidor	Fixos na extremidade esquerda da tela são fatais para o player.
Simples	Aqueles que somente possuem a movimentação padrão na horizontal.
Canhão	São canhões que atiram projéteis na direção do jogador.
Mola	Possuem molas no topo e elas impulsionam o jogador para cima.
Campo Intermitente	Elétrico Dois objetos que, ocasionalmente, ligam e disparam energia entre eles. Quando desligados são inofensivos.

Tabela 2: Obstáculos existentes no jogo

Os diferentes obstáculos do jogo servem como empecilho ao jogador, de forma que o mesmo deve desviar deles para prosseguir na fase; caso ocorra uma colisão entre o jogador e a maioria desses objetos, o personagem retrocede uma distância fixa, com uma exceção: quando o jogador estiver com o poder de invencibilidade ativo. Vale ressaltar que quando o jogador atinge um obstáculo perseguidor, a partida é encerrada. Veja a Tabela 2 para uma explicação detalhada de cada um dos obstáculos.

5. Método de Geração de Cenários Procedurais Adaptáveis

Como citado anteriormente um ponto de extrema importância no desenvolvimento de um título do gênero Endless Runner é garantir que o jogo transmita a sensação de que cada partida é única, despertando a curiosidade do jogador e incitando-o a continuar jogando. Para esse fim, foram utilizadas técnicas de GPC na geração dos cenários.

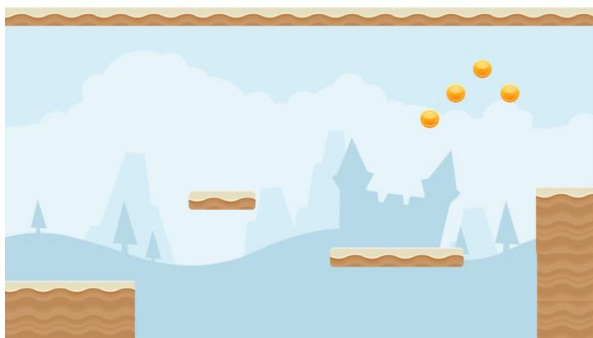


Figura 2: Um segmento de fase fácil.

Inicialmente foram criados, manualmente, segmentos de fase, cada segmento possuindo a largura equivalente a uma tela do dispositivo. No caso, como a plataforma alvo é iOS, a largura é igual a de um iPhone ou iPad. Isso significa que foi necessário produzir diferentes pedaços de cenário, distribuindo os obstáculos, power ups e moedas. No total foram criados 13 segmentos de fase distribuídos em diferentes categorias de dificuldade: fácil, normal e difícil. A quantidade de segmentos de fase que foram produzidas define a diversidade dos cenários que poderão ser gerados. O sistema que carrega as fases foi desenvolvido de tal forma que permite a adição ou remoção desses segmentos sem que nenhuma linha de código seja modificada.

Após essa etapa, todos os segmentos foram avaliados individualmente pelos autores. Nessa avaliação, os autores jogaram, testaram cada um dos segmentos e, por fim, os distribuíram em uma das categoria de dificuldade; também foi possível verificar se algum segmento produzido era impossível de ser superado pelo jogador, devido à organização dos elementos no cenário. Se esse fosse o caso, o segmento em questão era modificado e testado novamente ou descartado. O término desse estágio resultou em um banco de dados com diversos segmentos de cenário, já divididos por dificuldade, prontos para serem utilizados como parte de uma fase (veja Figuras 2, 3 e 4).

Durante o jogo, são constantemente analisadas informações relacionadas ao jogador e ao seu estilo de jogar a fim de determinar se o mesmo está com facilidade ou dificuldade para avançar na fase; com esses dados é possível definir o desempenho atual do jogador. Para eleger qual será o próximo pedaço de cenário, são considerados dois fatores: o desempenho atual do jogador e a dificuldade que o próximo segmento deve possuir.

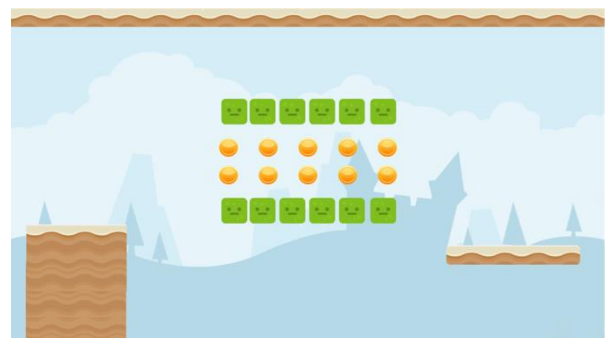


Figura 3: Um segmento de fase normal.

Todos os pedaços de fase são divididos por dificuldade em diferentes filas, uma fila para cada dificuldade, que são ordenadas de duas maneiras: na primeira ordenação, macro, todos os segmentos da mesma dificuldade são embaralhados entre si; a segunda ordenação, micro, só ocorre após a primeira

ordenação ter sido concluída e é conforme a quantidade de vezes que cada segmento já foi escolhido, de forma que quanto maior for esse número, mais ao final da estrutura de dados ele se encontra; Isso resulta em filas que possuem os pedaços de fases que não foram selecionados no início e os que já foram no fim.

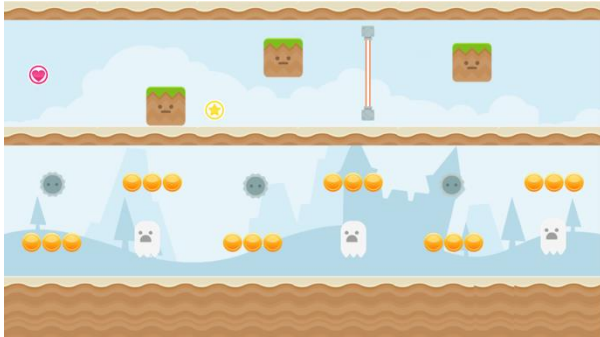


Figura 4: Um segmento de fase difícil.

O algoritmo seleciona o primeiro pedaço de fase existente na fila da dificuldade correspondente, então incrementa a quantidade de vezes que ele foi selecionado e reordena a mesma conforme descrito anteriormente. Por exemplo: o usuário está com o desempenho avaliado em 2 e foi detectado que ele está com facilidade para continuar jogando, o algoritmo determina que o próximo segmento deve ter dificuldade 3, então seleciona o primeiro elemento da fila de dificuldade 3, reordena-a e adiciona esse segmento na fase atual.

A ideia principal é criar cenários procedurais que coincidem com a habilidade do jogador, essa habilidade é armazenada numa variável chamada de *userHability*. Para isso, foi implementado um sistema que pontua o jogador conforme ele progride na fase. Ao instalar o jogo, o player possui sua *userHability* igual à 0 (zero). Essa variável é salva localmente no dispositivo, para manter a persistência da habilidade do jogador entre partidas, prevenindo o caso da *userHability* ser sempre zero no início de cada fase.

Toda vez que o usuário completar um segmento de fase, seu desempenho nesse segmento é avaliado. Se ele conseguiu completar a porção de cenário, sua *userHability* é incrementada; se ele conseguiu adquirir todas as moedas existentes, sua *userHability* também é incrementada; mas caso ele não tenha adquirido todas as moedas, sua habilidade é decrementada gradualmente conforme o número de moedas encontradas, quanto menor for esse número, maior será o decremento. Também existe a redução da *userHability* se o player for atingido por um obstáculo, cada obstáculo decrementa uma certa quantidade.

A *userHability* possui mais uma utilização: para definir a dificuldade da nova porção de fase que deve ser carregada. Cada segmento é classificado entre fácil, médio e difícil. Foi desenvolvido um sistema que gera a dificuldade que o segmento deve ter conforme a

atual *userHability* do jogador. Dessa forma, com o sistema de pontuação do desempenho do jogador juntamente com a geração procedural de segmentos de fase, é possível criar cenários que correspondam com o nível de habilidade do jogador, ou seja balanceados, ao mesmo tempo que são únicos, interessantes e desafiadores.

6. Avaliação

Para avaliar o método de GPC desenvolvido neste artigo, foi conduzido um experimento onde pessoas deveriam jogar o jogo *Dashing Aliens* e, posteriormente, responder um questionário. Este questionário foi criado e disponibilizado através de um formulário online. O formulário que os participantes responderam era composto por 8 afirmações, cada qual com cinco opções de resposta no formato Likert [Likert 1932].

Tarefa	Descrição
Tarefa 1	Criar um usuário
Tarefa 2	Acessar as telas de tutorial.
Tarefa 3	Jogar uma partida até falhá-la.

Tabela 3: Tarefas propostas para a primeira parte da avaliação.

A avaliação foi dividida em duas partes e realizada com um grupo de 10 pessoas, com idades variando entre 19 e 25 anos. Para cada pessoa foi entregue um dispositivo móvel do modelo iPhone, explicada a proposta do jogo e realizada uma demonstração, por parte dos autores, dos principais conceitos de *gameplay*; os participantes também foram instruídos a acessarem as telas de tutorial disponíveis dentro do próprio jogo. Na primeira parte da avaliação foi pedido para os participantes executarem 3 tarefas (mostradas na Tabela 3) dentro do jogo por, no mínimo 3 vezes. Todos os jogadores executaram as tarefas descritas na Tabela 3 no mínimo 3 vezes, alguns decidiram jogar mais do que o necessário.

Tipo de avaliação	Afirmação
Aceitação do jogo.	1 - O jogo é divertido.
Geração procedural de conteúdo.	2 - O jogo propõe desafios.
Geração procedural de conteúdo.	3 - Os desafios do jogo não são extremamente fáceis nem difíceis, eles se adaptam conforme a habilidade do jogador.
Aceitação do jogo.	4 - Os cenários do jogo possuem diversidade.
Aceitação do jogo.	5 - Cada cenário aparenta ser único.
Aceitação do jogo.	6 - O número de obstáculos no jogo é o

	suficiente para propor desafios.
Aceitação do jogo.	7 - O número de poderes no jogo é suficiente para deixá-lo interessante.
Geração procedural de conteúdo.	8 - O jogo possui “replay value” (o valor de entretenimento que faz com que um jogo seja jogado mais de uma vez).

Tabela 4: Afirmções do questionário.

Para a segunda parte da avaliação, cada um dos participantes foi conduzido a responder um questionário contendo 8 afirmações, veja Tabela 4. As afirmações foram orientadas para avaliar a aceitação do jogo (afirmações 1, 4, 5, 6 e 7) e o método de geração procedural de conteúdo utilizado (afirmações 2, 3 e 8).

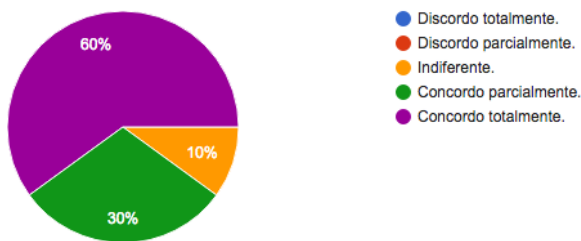


Figura 5: Resultados da afirmação 2.

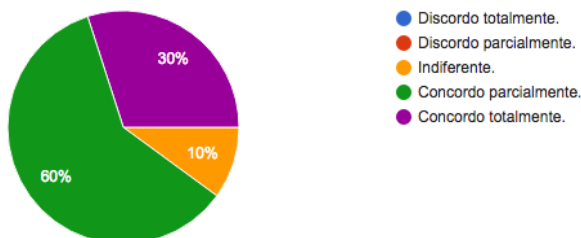


Figura 6: Resultados da afirmação 3.

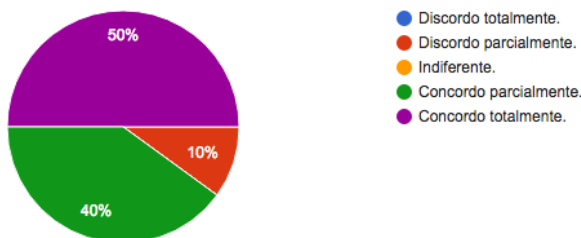


Figura 7: Resultados da afirmação 8.

Analisando os resultados obtidos (apresentados nas Figuras 5, 6 e 7) na avaliação do método de geração procedural de conteúdo, é possível

verificar que 90% dos entrevistados concordam que o jogo propõe desafios; dessa porcentagem, 60% concorda totalmente enquanto que o restante concorda parcialmente (veja Figura 5). Similarmente, na Figura 6, 90% dos participantes concordam que os desafios do jogo se adaptam conforme a habilidade de cada jogador, significando que o método de GPC está gerando cenários balanceados; dessa porcentagem, 30% concorda plenamente e 60% concorda parcialmente. Nos dois casos, Figuras 5 e 6, 10% dos jogadores é indiferente às afirmações. Os resultados referentes à última afirmação utilizada para a avaliação do método de GPC, a afirmação 8, demonstram que, novamente, 90% dos participantes concordam que o jogo possui “replay value”; para essa afirmação as porcentagens de cada um dos tipos de concordâncias diferem, sendo 50% para concordo totalmente e 40% para concordo parcialmente; finalmente, 10% dos entrevistados discordam parcialmente da afirmação, veja Figura 7. Através desses resultados (Figuras 5, 6 e 7) verifica-se que o método de geração procedural de conteúdo desenvolvido neste artigo atinge o objetivo estipulado pelos autores, sendo este o de gerar cenários adaptáveis que correspondem as habilidades de cada jogador resultando em fases balanceadas e que provêm desafios tanto a jogadores iniciantes quanto experientes.

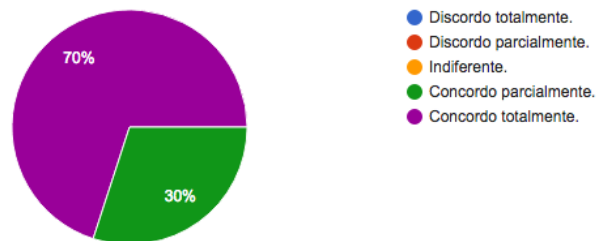


Figura 8: Resultados da afirmação 1.

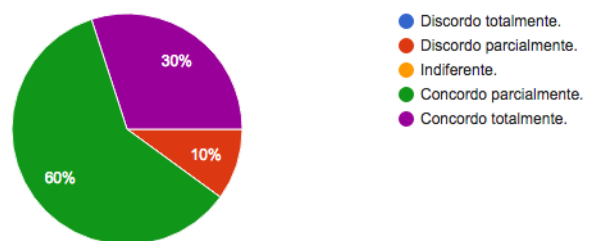


Figura 9: Resultados da afirmação 4.

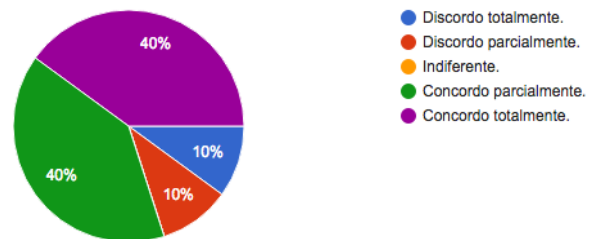


Figura 10: Resultados da afirmação 5.

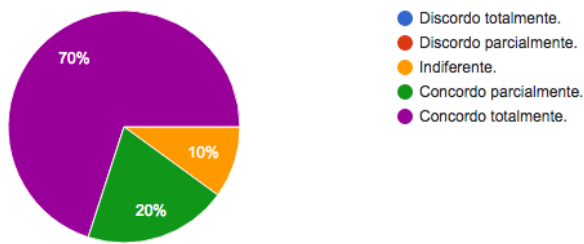


Figura 11: Resultados da afirmação 6.

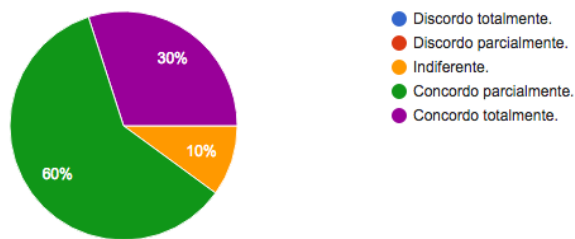


Figura 12: Resultados da afirmação 7.

Após examinar os resultados da avaliação da aceitação do jogo (apresentados nas Figuras 8, 9, 10, 11 e 12) verifica-se que 100% dos entrevistados concordam que o jogo é divertido, 70% concordando totalmente e 30% parcialmente, veja Figura 8. 90% dos jogadores concordam que os cenários do jogo possuem diversidade, sendo 30% totalmente e 60% parcialmente; os 10% restantes discordam parcialmente, veja Figura 9. Conforme a Figura 10, 80% dos participantes concordam que cada cenário aparenta ser único, sendo 40% tanto para totalmente quanto para parcialmente; os 20% restantes estão divididos igualmente entre discordo parcialmente e totalmente. Finalmente, por intermédio das Figuras 11 e 12, percebe-se que 90% dos jogadores concordam que o número de obstáculos e power ups é suficiente para propor desafios e tornar o jogo interessante: no caso dos obstáculos, 70% concordam totalmente e 20% parcialmente; enquanto que no caso dos power ups 30% concordam totalmente e 60% parcialmente; nos dois casos os 10% restantes são indiferentes. Com os resultados da aceitação do jogo, é possível afirmar que a média de aceitação dos jogadores foi de 90%, percentual alto, confirmando que o jogo é divertido, interessante, possui diversidade tanto de obstáculos e power ups quanto de cenários gerados pelo método de GPC.

7. Conclusão

Através de GPC é possível criar cenários infinitos que sejam interessantes ao jogador e que forneçam ao mesmo um desafio que não é baixo ao ponto do usuário sentir que está fácil, mas que também não é alto a ponto do usuário sentir que o jogo é injusto. De forma que tanto jogadores iniciantes quanto experientes possam jogar o mesmo jogo. Com isso é

possível abranger uma maior gama de usuários, pois não existe um nível fixo de habilidade requerido pelo jogo, ao invés disso ele se adapta conforme as habilidades de cada um de seus jogadores.

Através da avaliação conduzida neste trabalho, foi possível comprovar a eficiência do método de GPC desenvolvido. Este possibilitou a geração de fases balanceadas conforme a habilidade de cada jogador. Entretanto, verificou-se a necessidade de aprimoramentos no âmbito da diversidade dos cenários gerados.

Como trabalhos futuros pretende-se criar novos segmentos de fase para cada uma das categorias de dificuldade, incrementando o número atual de 13 segmentos por dificuldade; o objetivo a ser alcançado, para cada categoria, é de 30 segmentos. Posteriormente, após a efetuação de tais modificações, será realizada uma nova avaliação, nos mesmos moldes da avaliação presente neste trabalho, afim de aferir se as modificações realizadas contribuíram para a geração de cenários com maior diversidade.

Agradecimentos

Os autores gostariam de agradecer todos os envolvidos na realização deste trabalho.

Referências

- Apple, 2016. OS X. It's what makes a Mac a Mac. Disponível em <<http://www.apple.com/osx/what-is/>> [Acesso em: 11 julho 2016].
- Apple, 2016. Swift 3 The powerful programming language that is also easy to learn. Disponível em <<https://developer.apple.com/swift/>> [Acesso em: 11 julho 2016].
- Apple, 2016. The world's most advanced mobile OS. In its most advanced form. Disponível em <<http://www.apple.com/ios/what-is/>> [Acesso em: 11 julho 2016].
- Apple Developer, 2016. Cocoa Core Competencies. Disponível em <<https://developer.apple.com/library/mac/documentation/General/Conceptual/DevPedia-CocoaCore/ObjectiveC.html>> [Acesso em: 11 julho 2016].
- Apple Developer, 2016. iOS Dev Center. Disponível em <<https://developer.apple.com/devcenter/ios/index.action>> [Acesso em: 11 julho 2016].
- Apple Developer, 2016. Xcode: The complete toolset for building great apps. Disponível em <<https://developer.apple.com/xcode/>> [Acesso em: 11 julho 2016].
- Cocos2D, 2016. Create amazing games for mobile. Disponível em <<http://www.cocos2d-swift.org/>> [Acesso em: 11 julho 2016].
- Csikszentmihaly, M., 1991. Flow: The Psychology of Optimal Experience. 1st ed. New York.

- Dahlskog, S., Togelius, J., 2012. Patterns and procedural content generation: revisiting Mario in world 1 level 1. In Proceedings of the First Workshop on Design Patterns in Games (DPG'12). New York, NY, USA.
- Jennings-Teats, M., Smith, G., Wardrip-Fruin, N., 2010. Polymorph: A Model For Dynamic Level Generation. In The AAAI Press.
- Likert, R., 1932. A technique for the measurement of attitudes. Arch. Psychol., vol 22, no. 140, pp. 1-55.
- Medeiros, R. J. V. d., Medeiros, T. F. V. d., 2014. Procedural Level Balancing in Runner Games. In Brazilian Symposium on Computer Games and Digital Entertainment. Porto Alegre, pp. 109-114.
- Smith, G., Gan, E., Othenin-Girard, A., Whitehead, J., 2011. PCG-Based Game Design: Enabling New Play Experiences through Procedural Content Generation. In Second International Workshop on Procedural Content Generation in Games.
- Smith, G., Whitehead, J., Mateas, M., 2011. Tanagra: A Mixed-Initiative Level DesignTool. In Proceedings of the Fifth International Conference on the Foundations of Digital Games.