Análise da Aplicação de Geração Procedural no Contexto de Desenvolvimento do Jogo "Caso Sombrio"

João Pedro de Lima Ramos
Departamente de Ciência da Computação
PUC Minas – Unidade São Gabriel
Belo Horizonte, Brasil
joaop_lr@hotmail.com

Matheus Viana Brasileiro
Departamente de Ciência da Computação
PUC Minas – Unidade São Gabriel
Belo Horizonte, Brasil
matheusbrasileiro07@gmail.com

Marcelo Nery
Departamente de Ciência da Computação
PUC Minas – Unidade São Gabriel
Belo Horizonte, Brasil
msnery@gmail.com

Paulo Henrique Costa Moreira
Departamente de Ciência da Computação
PUC Minas – Unidade São Gabriel
Belo Horizonte, Brasil
thonlyph@gmail.com

Yuri Lincoln de Oliveira Couto
Departamente de Ciência da Computação
PUC Minas – Unidade São Gabriel
Belo Horizonte, Brasil
ylocouto@gmail.com

Figura 1. Processo de desenvolvimento da geração procedural do jogo "Caso Sombrio" — da prototipação em papel até a arte final

***Abstract*— Por mais que tenha começado como uma forma de compressão de dados, a geração de conteúdo de forma procedural ainda é utilizada, pois tem o potencial de criar novas experiências para o jogador sem muito esforço da área de arte e programação. Esse método, no entanto, traz outros desafios, já que sua natureza aleatória interfere com diversos elementos de um jogo, inclusive, seu balanceamento. Neste artigo são apresentadas as técnicas de geração procedural aplicadas no jogo "Caso Sombrio", um *runner* onde todos os cenários são construídos de modo a manter o jogador no estado de *flow*.**

***Palvras-chave:* geração procedural; balanceamento; desenvolvimento de jogos.**

I. INTRODUÇÃO

Métodos de geração procedural não são uma novidade na indústria dos jogos digitais. Por mais que tenha começado como uma forma de compressão de dados [1], a geração de conteúdo de forma procedural avançou nos últimos anos e hoje existem diferentes técnicas para criação de conteúdo *online* e *offline*, desde a construção automatizada de cenários, NPCs (personagens não jogáveis), mecânicas, inteligência artificial, itens, entre outros [2].

Um dos exemplos clássicos é o jogo *Rogue* [3], de 1980, que já utilizava algumas técnicas simples para criar diversos elementos como *dungeons*, personagens e cidades. A versatilidade advinda da geração procedural é fruto da forma como essa técnica funciona, já que pode utilizar diversos processos para criar, de maneira quase autônoma [4], uma gama de elementos diferenciados [5].

Uma das maiores preocupações com técnicas procedurais é ter controle em relação ao conteúdo que está sendo criado, principalmente se o desenvolvedor desejar incluir técnicas de adaptação dinâmica [2]. Para a maior parte dos conteúdos gerados automaticamente, o resultado final não influencia tanto no *design*. Porém, na construção de cenários é indispensável que este processo siga parâmetros pré-estabelecidos para que o conteúdo gerado funcione tão bem quanto feito manualmente.

Contudo, esse processo demanda uma análise individualizada para cada tipo de problema de game design. Na seção seguinte serão abordadas algumas destas questões e soluções sugeridas.

II. TRABALHOS RELACIONADOS

A abrangência da geração procedural traz, pela generalidade de suas aplicações, alguns desafios [6]. Suas técnicas são aplicadas em diferentes módulos de desenvolvimento de um jogo digital, variando da construção de cenários [3] até a automatização de áudio [7]. Desse modo, não existe uma metodologia única para geração procedural [8] que abrange todos os problemas possíveis de modo genérico. Logo, cabe a cada desenvolvedor abordar essa técnica da forma que mais beneficie a experiência que deseja criar.

Devido às diversas formas de tratar os conteúdos procedurais, existem várias maneiras de classificá-los como foi tratado em [9]. Já em [8], os diferentes métodos de geração procedural são classificados levando 13 elementos em consideração, sendo que os mais pertinentes para este projeto foram:

- *Entrada:* o que o algorítimo precisa receber para funcionar?
- *Complexidade:* o quão complexo é esse método?
- *Condição de parada*: como o algorítimo sabe quando parar e o que ele faz nesse momento?
- *Controle:* o quão bem o desenvolvedor consegue influenciar o resultado final?
- *Tempo de execução:* esse algorítimo é rápido o bastante para ser utilizado durante o jogo?
- *Variação:* o quão variado são os conteúdos gerados por este método?

Dentre os desafios provenientes da geração procedural, um dos mais presentes é garantir que o conteúdo que está sendo criado seja interessante o bastante para manter o jogador engajado [10]. Isso pode ser difícil, já que a natureza aleatória dessa técnica pode produzir variação de qualidade no jogo dependendo da forma como é aplicada. Como exemplo, no jogo *No Man's Sky* o jogador pode explorar um universo composto por planetas procedurais. Contudo, o jogo recebeu uma série de críticas da mídia especializada [11], pois, por mais que a diversidade de elementos criados fosse considerável, isso não afetava as experiências que o usuário tinha no decorrer do jogo, o que, a longo prazo, gerava desinteresse por parte dos jogadores.

Em casos onde há uma necessidade de balanceamento preciso e contínuo, como é a situação de "Caso Sombrio", é comum que partes sejam criadas manualmente para que os algoritmos as utilize no momento de gerar as fases [5]. A utilização de partes pré-montadas, porém, pode provocar repetições indesejadas. Como o ser humano é treinado em notar padrões[12], caso as diferentes peças utilizadas pelo gerador não sejam colocadas em um intervalo adequado, o jogador pode acabar reparando em padrões de repetição. Esse tipo de problema não só afeta o balanceamento, como também pode prejudicar a imersão do usuário.

Alguns jogos tentam resolver esse problema misturando o conteúdo feito manualmente com alguns elementos aleatórios procedurais. Como apontado por [13], em *Spelunky* (jogo onde o personagem deve explorar uma série de cavernas criadas proceduralmente) as fases são geradas a partir de partes feitas pelo desenvolvedor, mas o posicionamento de outros elementos em cada cena — como o dos inimigos — é feito por um algoritmo. Logo, as repetições são menos percebidas pelo jogador.

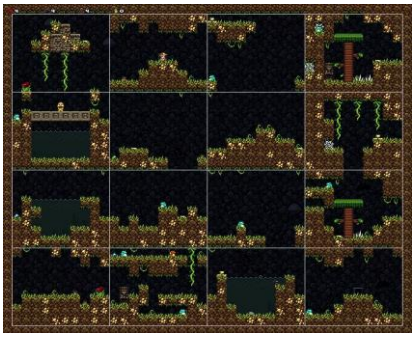

Figura 2. Exemplo de uma fase do jogo *Spelunky*, destacando a forma como ela é formada proceduralmente por diferentes partes que foram criadas de forma manual. Imagem extraída de [14].

Assim como já foi documento em [15], uma das melhores formas de manter o jogador engajado é sustentando o equilíbrio entre desafio e habilidade individual. Logo, é importante que isso seja levado em consideração para que o resultado final esteja devidamente balanceado. Aplicar essa teoria em conjunto com a geração procedural, no entanto, não é tão simples, pois, para resolver esse problema não basta dar ao gerador as peças que serão utilizadas. Portanto, para garantir que o resultado final seja balanceado, é importante, também, controlar quais partes estão sendo utilizadas e em quais momentos.

Por conta de todos esses fatores mostrados até agora – e outros que ainda serão apresentados no decorrer do texto – o objetivo desse artigo é analisar como eles podem ser abordados no desenvolvimento de um jogo. Para isso, utilizaremos como foco de estudo o desenvolvimento do jogo "Caso Sombrio".

As próximas partes do trabalho estão assim organizadas: na seção III serão apresentadas todas as características do jogo "Caso Sombrio"; na IV será mostrada a metodologia utilizada em sua criação (passando por como suas fases são geradas, os elementos que as compõe e como são balanceadas); e, tanto na V como na VI, os resultados e conclusões desse trabalho.

III. DESIGN DO JOGO "CASO SOMBRIO"

Dentre os pré-requisitos do desenvolvimento de "Caso Sombrio", foram propostos os seguintes elementos:

- Gênero: *infinity runner*;
- Plataforma: PC;
- Tempo de desenvolvimento: 10 meses;
- Precisa apresentar a mecânica de "siga o mestre", onde um personagem principal, controlado pelo jogador, vai acumulando outros personagens que o acompanham com atraso.

Além dos requisitos acima, ainda adicionamos mais uma mecânica principal ao jogo, que foi a de "campo de visão limitado".

O resultado final foi um *infinity runner* onde o jogador controla um personagem que corre constantemente da esquerda para a direita, além de poder pular, usar *dash* (investida para frente) e *drop dash* (investida para baixo). No início do jogo, o campo de visão disponível é pequeno, mas, durante o *gameplay*, o jogador encontrará sombras que, quando coletadas, aumentam essa área visível e começam a seguir o personagem principal.

Figura 3. Demonstração de como o campo de visão varia de tamanho dependendo da quantidade de sombras coletadas.

Figura 4. *Gameplay* de "Caso Sombrio".

Essas sombras respondem à todos os comandos dados pelo jogador, porém, com um certo atraso. Além disso, elas são imunes a todos obstáculos físicos, mas são destruídas ao colidir com a luz que entra através de buracos nas paredes. Quando uma sombra é destruída, o campo de visão conquistado ao coletá-la é perdido. Por fim, o jogo só termina quando o personagem principal entra em contato com um obstáculo físico.

"Caso Sombrio" se passa por 3 regiões principais: a Mansão, o Túnel e a Ruína (ilustradas na figura 5). Cada uma dessas partes possui características específicas, e por se tratar de um *infinity runner*, aparecem diversas vezes de forma intercalada.

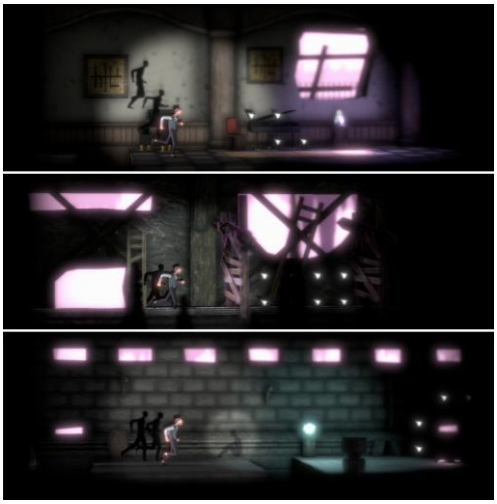

Figura 5. De cima para baixo: Mansão, Túnel e Ruína.

O jogo ainda apresenta outros elementos - como *power ups* e coletáveis relacionados ao seu enredo – os quais não serão aprofundados durante este artigo.

A. *Jogos Relacionados*

Durante o desenvolvimento de "Caso Sombrio", foram estudados outros jogos que possuem elementos semelhantes. O jogo *Zombie Tsunami* (figura 6), por exemplo, além de ser um *infinity runner*, também possui a mecânica de "siga o mestre", o que o tornou um ótimo material de estudo. Além disso, elementos como o intervalo entre a aparição de *power ups* e sua curva de dificuldade foram observados e serviram de guia para certos setores de "Caso Sombrio".

Figura 6. Em *Zombie Tsunami*, o jogador comanda um grupo de zumbis que deve sobreviver o maior tempo possível enquanto desviam de obstáculos e capturam humanos. A mecânica de "sigao mestre" está presente, pois o usuário só tem controle sobre um único zumbi, sendo que os restantes imitam o movimento dele após um período de tempo.

Em relação à mecânica de "campo de visão limitado", não foi encontrado outro jogo que já estivesse aplicando-a da mesma maneira em um *infinity runner*. Por essa razão, para compreender como algumas obras utilizam a dinâmica entre luz e escuro para amplificar a experiência do jogador, jogos do gênero suspense e terror (como o da figura 7) foram estudados.

Figura 7. Demonstração de como a luz é utilizada no jogo *Slender: The Eight Pages.* Esse tipo de iluminação, por limitar o que pode ser visto em cena, tem o potencial de gerar uma sensação de tensão no jogador (já que a qualquer momento algo pode estar escondido na parte escura do cenário). Logo, assim como apontado por [16], a maneira como a iluminação é aplicada em um jogo é fundamental para moldar como jogador irá percebe-lo.

IV. METODOLOGIA

A. *Sistema de Geração Procedural*

Depois que foi decido como funcionariam as mecânicas em "Caso Sombrio", deu-se início ao planejamento do sistema de geração procedural que seria utilizado. Para isso, foi feita uma pesquisa sobre outros trabalhos desenvolvidas na área.

Na obra apresentada em [17], é argumentado que analisar os elementos individuais de um jogo, de maneira isolada, não é uma boa forma para classificar a experiência que o jogador terá, já que, na verdade, ela é fruto das interações destes vários elementos entre si. Logo, é introduzido o conceito de "padrões", que consistem no agrupamento destes elementos em conjuntos que, quando abordados pelo jogador, produzem a experiência de *gameplay* desejada pelo desenvolvedor.

Já em [18], foram utilizados algorítimos para ajustar o conteúdo procedural a diferentes tipos de jogadores. O resultado final foi que 60% dos entrevistados preferiram as fazes ajustadas em detrimento das que foram geradas de forma puramente aleatória. Portanto, é possível inferir a importância de se ter formas de adequar o gerador procedural ao nível de dificuldade apropriado ao jogador.

A partir do que foi pesquisado, foram estipulados alguns requisitos que o gerador precisaria cumprir para que "Caso Sombrio" funcionasse de forma apropriada. Dentre eles estavam:

- Ser capaz de gerar fases à partir de módulos (padrões) pré-montados;
- Ter a capacidade de diferenciar esses módulos em vários níveis de dificuldade;
- Criar fases baseado em uma curva de dificuldade criada pelos desenvolvedores (dessa forma adequando a fase gerada ao nível esperado do jogador);
- Conseguir gerar conteúdo em tempo real, e por quanto tempo for necessário;
- Poder inserir elementos que não fazem parte dos módulos criando assim uma maior diversidade de conteúdo;

O resultado final foi um gerador que, em tempo real, cria as fases à partir de módulos feitos manualmente. Cada um desses módulos foi classificado entre 8 níveis de dificuldade diferentes, e são dispostos seguindo o sistema de balanceamento que será melhor detalhado na seção IV.

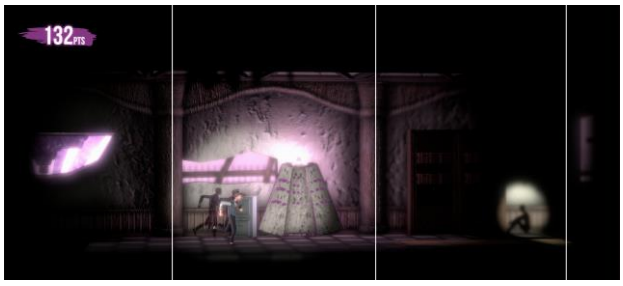

Figura 8. Essa imagem apresenta 4 módulos diferentes encaixados para formar um trecho da Mansão.

Como todo o balanceamento gira em torno da forma como os módulos são montados, foi decidido que a maneira mais adequada de aplicar elementos que não fizessem parte desses padrões – como proposto no quinto requisito – seria se esses não interferissem na dificuldade do jogo. Dessa forma, para esta parcela independente ficou responsável os elementos de *background* e *foreground*, que, por mais que não afetem o *gameplay*, aumentam a imersão do jogador ao disfarçar as repetições que podem acabar acontecendo em um gerador que funciona a partir de módulos pré-montados (como foi mencionado na seção I).

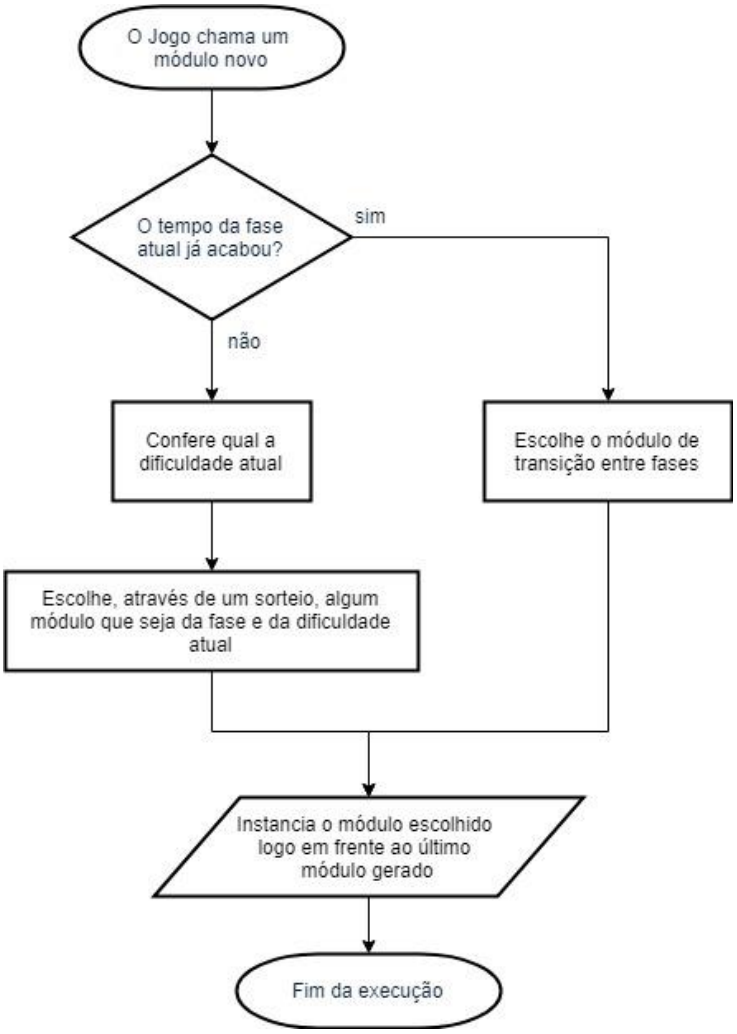

Figura 9. Fluxograma de funcionamento do gerador utilizado.

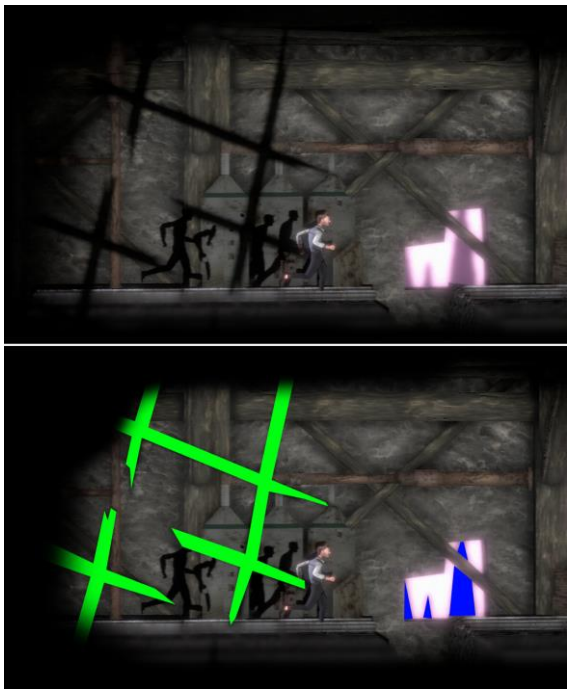

Figura 10. Em verde, os elementos do *foreground*. Em azul, os de *background*.

Assim como apontado em [12], uma das melhores maneiras de manter um jogador interessado, é tirando proveito de sua curiosidade. Por esse motivo, a duração de cada fase em "Caso Sombrio" é 30 segundos. Esse período curto de tempo permite que o jogador experimente a fase atual, e, antes de começar a se desinteressar, seja introduzido a um novo contexto para recapturar sua atenção. Como jogo fica cada vez mais difícil, a cada 30 segundos novas combinações de obstáculos são introduzidas. Logo, mesmo se tratando de apenas 3 fases, ainda sim é possível surpreender o jogador com experiências ainda não vistas.

B. Planejamento dos Módulos

Com o intuito de otimizar o processo, toda a primeira etapa de criação dos módulos foi feita com protótipos de papel. Como destacado por [19], a utilização dessa técnica permite que alguns setores do jogo sejam testados sem a necessidade de elementos mais complexos (como a programação). Dessa forma, o processo de prototipação pode ocorrer de forma mais rápida e barata [20], além de possibilitar a identificação e correção de eventuais falhas ainda nos primeiros estágios do desenvolvimento.

Antes da montagem dos módulos, primeiro foram criados – de forma individual - os elementos que os compõe. Estes componentes podem ser agrupados em 4 grupos:

- **Chão** (define onde estão os buracos que o personagem principal precisa evitar);
- **Parede** (define por onde entra a luz que as sombras precisam evitar);
- **Coletáveis**;
- **Obstáculos**.

Para padronizar o tamanho desses elementos, foi adotado que o personagem principal teria 2 metros de altura e que as dimensões dos módulos seriam 7 metros de altura por 8 de comprimento. Dessa forma, todas as partes foram planejadas com essas proporções em mente.

Seguindo o processo estipulado pelo grupo, os primeiros componentes criados foram os diferentes tipos de chão, sendo que as dimensões adotas para eles foram de 1 metro de altura por 8 metros de comprimento. Para que o resultado final conseguisse gerar múltiplas experiências no decorrer do jogo, foram montadas versões que exigissem diferentes reações do jogador (como um pulo longo ou vários pulos curtos). Ao todo, foram criadas 18 variações diferentes de chão.

Em seguida, foram planejadas as paredes. Seguindo essa mesma ideia de criar elementos que fizessem o jogador agir de forma diferente, os buracos colocados possuem tamanhos e posições variadas. Na parede D4, presente no lado esquerdo na figura 12, o jogador precisa dar 3 pulos consecutivos para evitar que alguma de suas sombras passe em frente aos focos de luz. Já no caso da parede C3, o jogador não deve pular ao final, se não uma de suas sombras entrará em contato com o buraco localizado logo acima de sua posição. Ao fim desse processo, foram criados 21 tipos de parede.

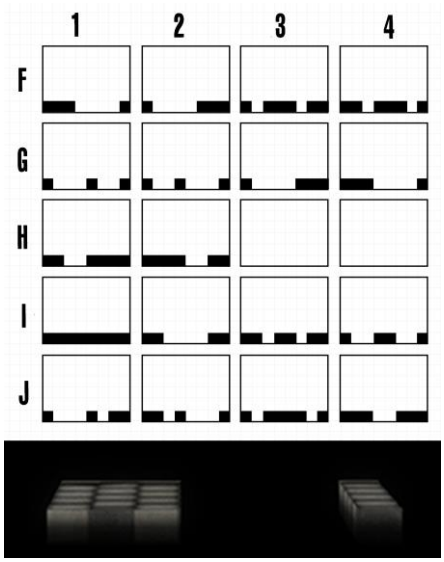

Figura 11. Na parte superior esta a representação inicial dos 18 tipos diferentes de chão. Em baixo, como a variante G4 ficou na versão final.

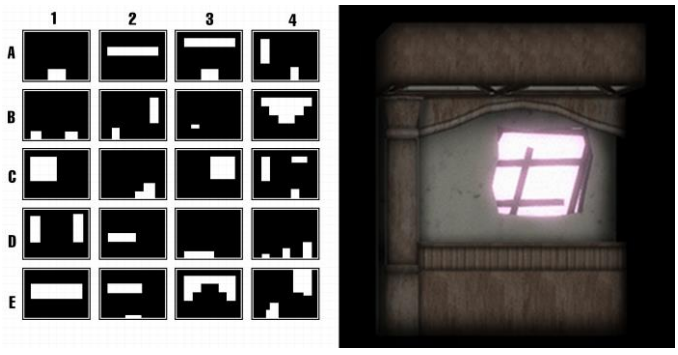

Figura 12. Na esquerda esta a representação inicial dos 21 tipos diferentes de parede. A direita, como a variante C3 ficou na versão final.

Sobre os coletáveis, existem 4 tipos diferentes em "Caso Sombrio": Fragmentos de Luz (aumentam a pontuação ao serem coletados), *Power Ups* (ativam efeitos temporários como intangibilidade e pulo duplo), Pistas (desbloqueiam trechos do enredo) e as Sombras (expandem o campo de visão). Por terem uma maior influência sobre o balanceamento, tanto os Fragmentos de Luz quanto as Sombras serão mais aprofundados na seção IV.

Por fim, como cada obstáculo é único de sua fase, eles foram criados e agrupados da seguinte forma:

Mansão:

- **Dama de ferro:** tenta pegar o personagem quando ele está passando em sua frente. É evitada com o *dash* e ocupa 4 metros de altura por 3 metros de comprimento;
- **Lustre:** emite pouca luz, logo pode ser visto mesmo quando o jogador possui um campo de visão reduzido. Ele começa a cair um pouco antes do personagem passar por debaixo dele. É evitado com o *dash* e ocupa 1 metro de altura por 3 metros de comprimento.

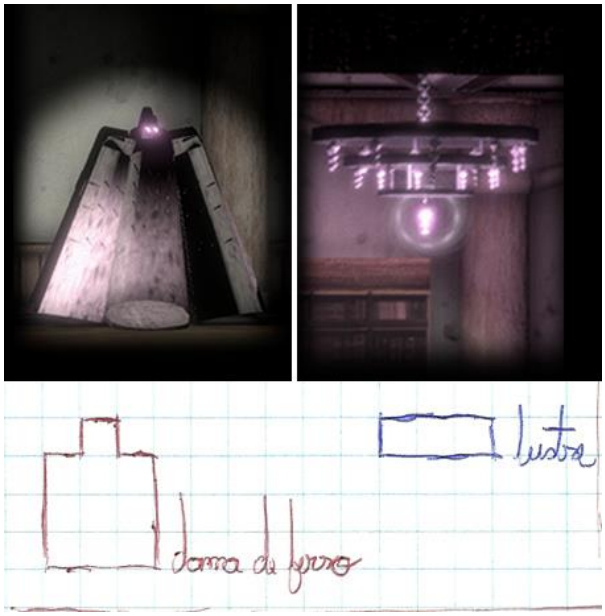

Figura 13. Assim como o restante dos elementos, os obstáculos - como a Dama de Ferro e o Lustre - também foram planejados durante o protótipo de papel.

Túnel:

- **Barreira de madeira:** tampa toda a passagem do personagem. Deve ser destruída com um *dash* e ocupa 5 metros de altura por 1 metro de comprimento;
- ***Minecart***: pode estar parado ou em movimento. Ocupa o caminho do protagonista e deve ser evitado com um pulo. Possui 1 metro de altura por 1 metro de comprimento.

Ruína:

- **Portão de ferro:** começa a fechar quando chega a certa distância do personagem principal, logo, precisa ser alcançado antes que bloqueie todo o caminho. Deve ser evitado com o *dash* e ocupa 6 metros de altura por 1 metro de comprimento;
- **Armadilha de espinhos:** é um botão que se ativa quando o personagem passa por cima dela. Pode ser evitado tanto com o pulo quanto com o *dash*. Ocupa 1 metro de altura por 1;
- **Bloqueio de madeira suspenso:** bloqueia todo o caminho do jogador, mas possui um bloco de pedra em sua base. Exige que o personagem pule e, logo em seguida, use o *dash* para quebrar sua parte de madeira. Enquanto o bloco possui 1 metro de altura por 1 metro de comprimento, a parte de madeira tem 4 metros de altura por 1 metro de comprimento metro de comprimento;

C. Montagem e classificação

Como indicado em [21], para que o *level design* funcione como o esperado, não basta apenas pensar em qual elemento está presente em cena, pois deve-se levar em conta, também, o seu posicionamento. Portanto, esse foi um dos princípios fundamentais utilizados durante a montagem dos módulos.

Na Figura 14, por exemplo, o jogador é incentivado a passar pelo botão utilizando um *dash*, pois, caso pule, sua sombra será destruída pela luz que entra através do buraco na parede. O mesmo não aconteceria, no entanto, se o botão estivesse mais à direita, já que o personagem principal iria poder pula-lo sem nenhum problema.

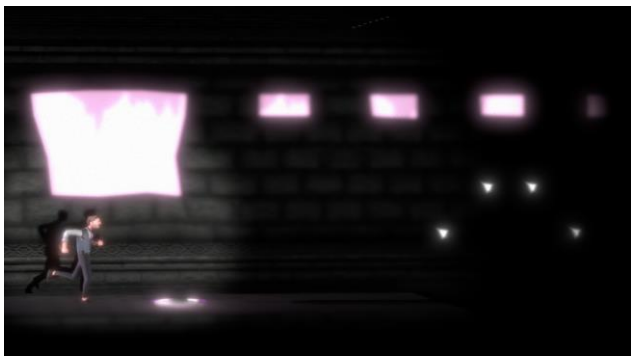

Figura 14. Demonstração de como a posição dos elementos afeta a experiência do jogador.

Outro exemplo presente ainda na mesma figura é em relação à posição dos fragmentos de luz. Como eles emitem uma aura luminosa e podem ser vistos a todo momento (inclusive quando o campo de visão está pequeno), o posicionamento desse coletáveis foi pensado de forma a guiar o jogador no caminho mais seguro.

Após a criação dos módulos, algumas fases foram montadas (ainda no papel). Esses primeiros testes, mesmo sendo adaptados para funcionar de maneira analógica, permitiram o ajuste de elementos importantes como a velocidade do personagem e a distância percorrida pelo *dash*. Além disso, foi detectado que alguns módulos estavam muito difíceis. Logo, uma parcela destes foi ajustada, e, o restante, descartado. Ao todo, foram criados 392 tipos diferentes.

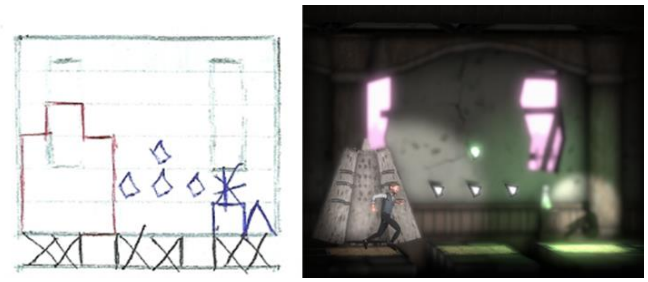

Figura 15. Versão de um módulo durante o protótipo de papel e sua versão final.

No projeto abordado em [5], a maneira utilizada para controlar o balanceamento do gerador procedural foi a classificação dos módulos em vários níveis de dificuldade. Isso permitiu que, durante a criação das fases, o desenvolvedor tivesse mais domínio sobre o quão difícil elas seriam. Logo, visando um resultado parecido, foi aplicada a mesma metodologia em "Caso Sombrio".

Durante a classificação dos módulos, não foi observado quais obstáculos exatamente estavam presentes em cena, e sim, quais ações o jogador deveria realizar para não ser atingido. Foi adotado esse método, pois, assim como mencionado na seção IV, a análise dos objetos de forma isolada nem sempre conclui de forma correta qual será a experiência do jogador ao se deparar com determinados elementos.

Na figura 16, por exemplo, está sendo demonstrado como 2 módulos diferentes foram verificados. No da esquerda, como o personagem precisa realizar diversos movimentos em sequência – além de ter que utilizar um dash enquanto está caindo -, o nível de dificuldade aplicado foi 6. Já o módulo da direita foi classificado como 2, já que com apenas um *dash* e um pulo é possível vencê-lo. Ao todo, foram adotados 8 níveis de dificuldade para a classificação dos módulos.

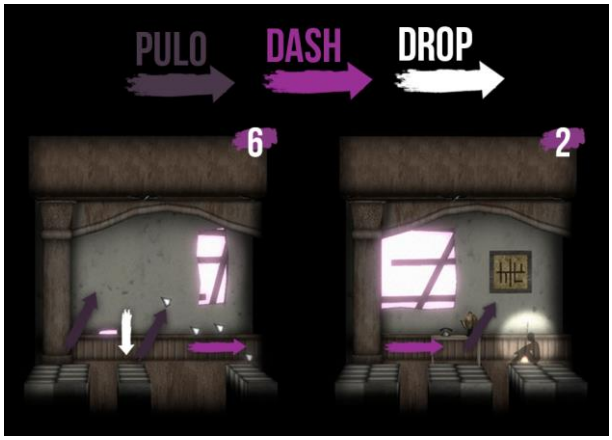

Figura 16. Demonstração de como os módulos foram classificados.

D. *Balanceamento*

No artigo [22] é levantada a hipótese de que não são os elementos "superficiais" - como os gráficos e a temática – que constituem um bom jogo, sendo que seus autores chegaram a essa conclusão após analisar alguns estudos, dentre eles, a pesquisa sobre o *Flow* [15].

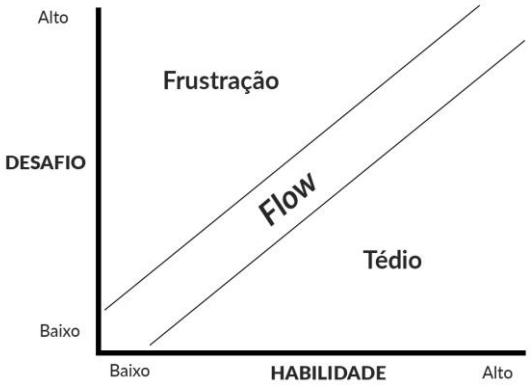

Figura 17. Gráfico que ilustra o equilíbrio entre dificuldade e habilidade proposto na teoria do *Flow* [15].

De acordo com *Csikszentmihalyi*, *Flow* é um estado mental onde um indivíduo está completamente imerso em determinada experiência [22], sendo que alguns dos principais critérios para atingilo são o equilíbrio entre desafio e a habilidade do jogador, além da presença de recompensas [15].

Visando garantir que fosse possível atingir o estado de *Flow* durante o *gameplay* de "Caso Sombrio", foi desenvolvida uma curva de balanceamento que visa controlar a dificuldade dos módulos colocados pelo gerador.

Como é possível observar na figura 18, por mais que o jogo se torne mais difícil com o decorrer do tempo, existem diversos pontos no decorrer da experiência onde a dificuldade é mais baixa. Esses momentos servem como recompensa para o jogador, pois permitem que ele "descanse" entre as partes mais difíceis, além de também fornecerem uma quantidade maior de fragmentos de luz (aumentando a pontuação).

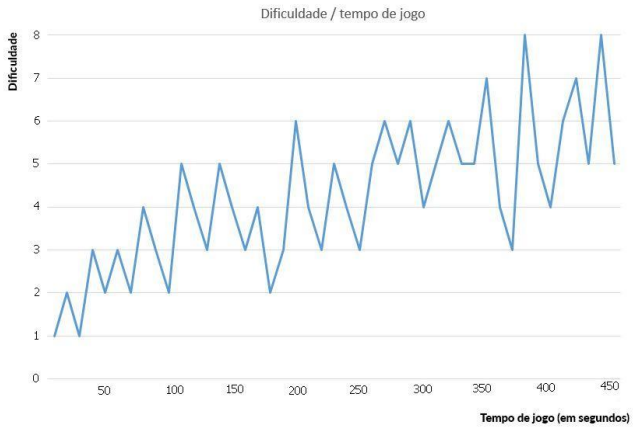

Figura 18. Gráfico que representa a variação de dificuldade a cada 10 segundos de jogo.

Além do mencionado acima, a variação da velocidade do personagem também contribui com a dificuldade de "Caso Sombrio". No início do jogo o protagonista está em sua velocidade mínima, o que permite que o jogador jogue de maneira confortável mesmo com um campo de visão pequeno. Com o decorrer do tempo, no entanto, a velocidade fica cada vez maior, o que acaba exigindo um tempo de reação preciso na hora de evitar obstáculos. Dessa forma, coletar sombras se torna cada vez mais

importante, pois, como o personagem esta ficando mais rápido, o campo de visão expandido permite que o jogador detecte possíveis perigos com maior antecedência e responda de forma ágil.

Outro elemento que precisou ser balanceado foi a frequência com que as sombras aparecem. Quando um módulo é gerado, o jogo decide se a sombra presente nele está ativa ou não. Logo, para garantir que o jogador não fique muito tempo com o campo de visão mínimo - mas que também não seja fácil atingir o máximo (5 sombras) - , a chance da sombra ser ativada é inversamente proporcional à quantidade que o jogador possui no momento em que o módulo em questão é colocado em cena (o que está mais detalhado na tabela 1).

TABELA I. CHANCE DE ENCONTRAR SOMBRAS

Sombras coletadas	Chances de encontrar nova sombra
0	100%
1	60%
2	50%
3	35%
4	20%
5	0%

Por fim, o último componente balanceado foi o funcionamento da mecânica de "campo de visão limitado", pois, como o personagem está sempre em movimento, é necessário garantir que mesmo com o campo de visão mínimo o jogo não seja frustrante para o jogador. Para isso, todos os obstáculos emitem um brilho quando estão próximos ao protagonista, além de que a borda direita dos buracos reflete a luz do personagem. Deste modo, o jogador é alertado que está próximo de um buraco mesmo sem ver onde ele começa, possibilitando que fique mais atento em relação ao obstáculo que está por vir.

Figura 19. Nessa imagem está destacada pelo círculo vermelho o reflexo da luz do personagem na borda direita do buraco.

V. TESTES FINAIS

Para os testes de verificação da qualidade da jogabilidade, foram selecionados vinte e seis participantes, sendo nove mulheres e dezessete homens. O jogo foi exposto em uma feira de jogos e os participantes que desejaram jogar foram acompanhados durante toda a seção de jogo, sendo observadas suas reações e, posteriormente, um pequeno questionário de perguntas foi aplicado. Cada seção de jogo durou, em média, cinco minutos.

Os participantes adotaram os seguintes critérios: jovens entre 16 e 30 anos, jogadores assíduos e alfabetizados, participando de modo voluntário.

A. *Coleta de dados*

A coleta de dados foi dividida em três etapas e ocorreu durante um período de um dia.

Na primeira etapa, o participante respondeu a um questionário demográfico que coletou informações referentes à idade, sexo, nível de escolaridade e experiência prévia com jogos. Todos os instrumentos utilizados foram aprovados pelo Comitê de Ética em Pesquisa. Cada participante selecionado assinou um termo de consentimento para participação na pesquisa. O mesmo ressalta que as informações obtidas serão confidenciais, sendo assegurado o sigilo sobre a participação, uma vez que os resultados serão sempre apresentados como retrato de um grupo e não de uma pessoa.

Na segunda etapa, o jogador jogava o jogo de forma livre, sem interferência dos avaliadores, sendo apenas observados e suas reações perante as ações no jogo anotadas.

Na terceira e última etapa, os jogadores respondiam algumas questões sobre sua experiência no jogo, conforme será apresentado na seção a seguir.

B. *Heurística de pesquisa*

Seguindo as recomendações de [23], que define dez princípios gerais (heurísticas) de design de interface do usuário para avaliação de usabilidade, definiu-se alguns pontos de interesse para avaliar a jogabilidade através do uso de geração procedural. As heurísticas observadas foram:

- (i) Visibilidade do estado do sistema;
- (ii) Liberdade e controle do usuário;
- (iii) Consistência e padrões, que remete a reconhecer ao invés de relembrar;
- (iv) Prevenção de erro;
- (v) Estética e design minimalista.

Nem todas as heurísticas eram de interesse nessa pesquisa de modo que somente as que diziam diretamente respeito a level design e game design foram selecionadas. Algumas questões colocadas aos jogadores foram: (i) "Você reconhecia os obstáculos facilmente?", (ii) "Está fácil controlar as ações do personagem?" e "O personagem responde rapidamente aos seus comandos?", (iii) "Você percebeu o que eram obstáculos de luz, buracos e coletáveis?" e "Você percebeu como diferenciar os obstáculos dos demais objetos através do sistema de cores usado?", (iv) "Você notou que os obstáculos eram perceptíveis mesmo no escuro?" e (v) "O jogo possui um

design limpo e fácil de identificar o que é necessário para jogar?".

C. *Análise dos dados*

A análise dos dados foi qualitativa. Nessa análise buscou-se identificar pontos positivos e negativos do jogo na visão das pessoas. Com base nesses pontos, os desenvolvedores chegaram a algumas observações importantes que guiaram alterações no design final do jogo:

- Por ser algo diferente, a primeira reação de muitos jogadores foi testar a mecânica de campo de visão limitado. Como o personagem começa de forma lenta, isso permitiu com que os jogadores experimentassem esse elemento de maneira mais gradual e tranquila, o que resultou em um rápida assimilação sobre como exatamente o jogo funciona;
- Alguns jogadores tiveram dificuldades em identificar se um obstáculo deveria ser evitado com o pulo ou com o *dash*. Para facilitar esse aspecto do jogo, a cor emitida pelos obstáculos foi diferenciada entre duas: azul para os que utilizam o *dash* e rosa para os que devem ser evitados com pulo;

Figura 20. Como para passar pela dama de ferro é necessário utilizar o *dash*, ela emite um brilho roxo. Ja o *minecart*, por necessitar que o personagem pule, emite um brilho rosa.

- Em certos trechos de maior dificuldade, foi relatado que alguns momentos estavam ficando muito estressantes. Em resposta a esse problema, sempre que um módulo de dificuldade 7 ou 8 é gerado, o seguinte passou a ser um módulo de dificuldade 3 ou menos (permitindo que o jogador tenha tempo para se recuperar de desafios avançados);
- Mesmo conseguindo diferenciar os *Power Ups* do restante dos coletáveis, alguns dos entrevistados não compreenderam quais poderes eles ganhavam ao coletá-los. Para deixar esse aspecto mais claro, agora é mostrado momentaneamente o nome do power up sempre que o jogador o coleta;
- Como muitos jogos do gênero *runner* apresentam apenas uma fase, a primeira impressão dos participantes era de que o mesmo era verdade para "Caso Sombrio". Isso foi considerado um problema, já que uma das principais características do jogo são os diferentes obstáculos entre cada região. Logo, para deixar claro para o jogador que existe uma progressão em "Caso Sombrio', foi colocada na *HUD* uma barra de progresso que é preenchida no decorrer da fase;
- Durante os testes, foi percebido que o campo de visão inicial era muito punitivo para alguns jogadores. Logo, mesmo que "Caso Sombrio" tenha o escuro como parte de seu mecânica central, a iluminação inicial foi aumentada para impedir que o jogador se frustre já nas primeiras partidas.

Figura 21. Acima, a caixa de texto com o nome do *power up* de intangibilidade. Em baixo, a barra de progresso da fase em que o jogador se encontra.

Ao fim dos testes, "Caso Sombrio" foi considerado um jogo difícil, porém justo. A mecânica de campo de visão limitado, mesmo sendo diferente, foi bem aceita, e a maioria dos jogadores elogiaram a variedade de desafios e declararam ter interesse em continuar jogando.

VI. CONCLUSÕES

Durante a criação de "Caso Sombrio", foi percebido como a utilização da geração procedural afeta diversas etapas do desenvolvimento de um jogo. Logo, está apresentado neste artigo alguns métodos que foram empregados para a aplicação de elementos procedurais da melhor forma possível.

Com base no que foi apontado neste trabalho, uma das principais conclusões é a importância de se estudar - desde o início – sobre como exatamente a geração procedural sera aplicada. Isso é de extrema importância, pois já que elementos procedurais exigem alguns cuidados específicos - como foi a situação dos módulos de "Caso Sombrio" -, planejar com antecedência evita que elementos criados no início do projeto tenham que ser adaptados no futuro para se adequarem ao método de geração escolhido (economizando, assim, tempo e dinheiro).

Ainda no tópico de planejamento, outra observação digna de destaque é sobre a importância de se reservar uma parte do período de desenvolvimento para a criação de protótipos. Esse fator foi fundamental no desenvolvimento do jogo, já que permitiu o teste de novas ideias de maneira rápida e barata. Além disso, foi possível planejar todo o funcionamento do gerador ainda durante o protótipo de papel, logo, quando chegou o momento de programá-lo, já era sabido exatamente como ele deveria funcionar, fato que agilizou o processo de desenvolvimento.

Por fim, conclui-se que a geração procedural é uma ferramenta que, quando usada com planejamento, tem o

potencial de contribuir com diversos aspectos de um jogo. Portanto, é esperado que esse estudo possa servir de referência para desenvolvedores interessados em utilizar essa técnica em seus projetos.

REFERÊNCIAS

[1] A. Amato, "Procedural content generation in the game industry," Game Dynamics, 2017.

[2] R. M. Smelik, T. Tutenel, R. Bidarra, and B. Benes, "A Survey on Procedural Modelling for Virtual Worlds," Computer Graphics Forum, 2014.

[3] B. P. de Castro, R. R. da Mota, and E. P. C. Fantini, "Level design on rogue-like games: An analysis of crypt of the necrodancer and shattered planet," XV Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2016.

[4] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," ACM Trans. Multimedia Comput. Commun. Appl., 2013.

[5] L. V. Carvalho, Átila V M Moreira, V. V. Filho, M. T. C. F. Albuquerque, and G. L. Ramalho, "A generic framework for procedural generation of gameplay sessions," XII Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2013.

[6] R. Lopes and R. Bidarra, "Adaptivity challenges in games and simulations: A survey," IEEE Transactions on Computational Intelligence and AI in Games, vol. 3, pp. 85–99, June 2011.

[7] M. C. Silva, F. M. G. França, and G. R. E. Cabral, "Construindo trilhas sonoras dinâmicas em jogos utilizando sistemas fuzzy," XIII Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2014.

[8] N. Oliveira and R. D. Seabra, "Towards a comprehensive classification for procedural content generation techniques," XV Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2016.

[9] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," ACM Trans. Multimedia Comput. Commun. Appl., vol. 9, pp. 1:1–1:22, Feb. 2013.

[10] R. Koster, Theory of Fun for Game Design. O'Reilly Media, Inc., 2nd ed., 2013.

[11] D. Heaven, "When infinity gets boring: What went wrong with No Man's Sky." *https://goo.gl/zRqq3i*, 2016. Online; acessado em 01 de Agosto de 2018.

[12] J. Schell, The Art of Game Design a Book of Lenses. AK Peters, 2014.

[13] D. Yu, Spelunky. Boss Fight Books, 2016.

[14] M. Brown, "How (and Why) Spelunky Makes its Own Levels." *https://www.youtube.com/watch?v=Uqk5Zf0tw3o/*, 2016. Online; acessado em 01 de Agosto de 2018.

[15] M. Csikszentmihalyi, Flow: The Psychology of Optimal Experience. Harper Perennial Modern Classics, 2008.

[16] E. C. da Silva and S. Nesteriuk, "Luz nos games: por uma abordagem interdisciplinar na indissociabilidade forma, conteúdo e função," XV Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2016.

[17] K. Compton and M. Mateas, "Procedural level design for platform games," Proceedings of the Second AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 2006.

[18] N. Shaker, G. Yannakakis, and J. Togelius, "Towards automatic personalized content generation for platform games," Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 2010.

[19] L. G. Roque, "Early game design rehearsal with paper prototyping," IX Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2010.

[20] M. M. Filho, I. V. Benicio, F. Campos, and A. M. M. Neves, "A importância da prototipação no design de games," XII Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2013.

[21] R. C. Braga and R. R. da Mota, "Análise de level design um estudo de caso do jogo super mario bros," XIV Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2015.

[22] R. C. R. dos Santos, K. de Geus, S. Scheer, A. Miquelin, S. R. Jr, and W. C. Godoi, "On player motivation and the appeal of games: An exploration of player motivation," XVI Simpósio Brasileiro de Jogos e Entretenimento Digital-SBGames, 2017.

[23] J. Nielsen, "10 Usability Heuristics for User Interface Design." *http://www.nngroup.com/articles/ten-usability-heuristics/*, 1995. Online; acessado em 10 de setembro de 2018.