UNIVERSIDADE FEDERAL DE RORAIMA
CENTRO DE CIÊNCIA E TECNOLOGIA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

RELATÓRIO DO TRABALHO FINAL DE COMPUTAÇÃO GRÁFICA

Bento 's Trial – O teste de virtude do Bento.

Boa Vista
Março de
2025

GABRIEL PEIXOTO MENEZES DA COSTA
LUIZ GUSTAVO DALLAGNOL CAVALCANTE

RELATÓRIO DO TRABALHO FINAL DE COMPUTAÇÃO GRÁFICA

Relatório para obtenção de nota na disciplina Computação Gráfica no semestre 2024.2, sob orientação do professor Dr. Luciano Ferreira Silva.

Boa Vista
Março de
2025

Sumário

1. INTRODUÇÃO

Este relatório foi elaborado para a avaliação da disciplina de Computação Gráfica no semestre 2024.2. Aqui, apresentamos o processo de desenvolvimento de um jogo criado na Unity, destacando seus principais mecanismos, desafios superados e as técnicas de computação gráfica aplicadas ao longo do projeto. Nosso objetivo é demonstrar o funcionamento do jogo e evidenciar como os conceitos estudados em sala foram incorporados de maneira prática.

Link para baixar os arquivos do jogo:
https://github.com/GugaCS50/ProjetoFinalCG2025_LuizGustavo_GabrielPeixoto

2. ROTEIRO DO JOGO

Nosso jogo se passa em um ambiente de montanhas geladas e acompanha a jornada de Bento, um jovem pássaro que está aprendendo a voar. No início, Bento ainda não tem a força necessária para alçar voo e depende exclusivamente de saltos bem calculados entre plataformas. O jogador precisa dominar a estratégia e o timing para superar os desafios, evitando perigos como espinhos e quedas fatais.

Conforme avança no jogo e supera os obstáculos, Bento gradativamente conquista a habilidade de voar, representando um sistema de progressão baseado no desempenho do jogador. Dessa forma, a jogabilidade reflete a jornada de crescimento e aprendizado do protagonista.

3. DETALHES DO JOGO

Desenvolvido na Unity, este jogo de plataforma 2D coloca o jogador no controle direto de Bento, permitindo que ele se mova, salte e escale. O sistema de física do jogo é gerenciado pelo Rigidbody2D e o Input System, garantindo uma jogabilidade responsiva e intuitiva.

Para garantir um gameplay preciso, utilizamos um sistema de detecção de solo baseado na função OverlapBox. Além disso, implementamos um sistema customizado de gravidade que acelera a queda, tornando os movimentos mais naturais e desafiadores. O objetivo é conduzir Bento até o final da fase, superando os desafios e evitando os perigos do cenário.

4. FERRAMENTAS UTILIZADAS

4.1 Unity 6

A Unity 6 foi a engine escolhida para o desenvolvimento do jogo devido à sua flexibilidade e suporte avançado para jogos 2D. Com ela, conseguimos integrar diversos recursos essenciais, como:

- **Input System:** Para o gerenciamento dos comandos do jogador.
- **Rigidbody2D:** Para a física do personagem.
- **Manipulação de gravidade personalizada:** Para um controle mais realista dos saltos e quedas.
- **Gerenciamento de assets:** Facilitando a criação e interação das mecânicas do jogo.

5. PRINCIPAIS TÉCNICAS DE COMPUTAÇÃO GRÁFICA APLICADAS

5.1 Parallax Scrolling

Para criar a ilusão de profundidade no cenário, utilizamos a técnica de Parallax Scrolling. Essa técnica permite que diferentes camadas do fundo se movimentam em velocidades distintas, reforçando a sensação de imersão. O efeito foi implementado por meio do script **BckgScrolling**, que ajusta dinamicamente a posição das sprites com base no movimento da câmera.

5.1.1 Organização das Camadas

As imagens de fundo foram organizadas conforme a profundidade:

- **Aurora Boreal (Figura 2):** Camada mais distante, com o menor valor de deslocamento.
- **Céu (Figura 3):** Camada intermediária, com um valor médio de deslocamento.
- **Montanhas (Figura 4):** Camada mais próxima, com o maior valor de deslocamento.

Essa disposição garante um efeito visual suave e realista, proporcionando maior imersão ao jogador.

5.1.2 Implementação e Uso do LateUpdate

Ao desenvolver a movimentação do cenário, optamos por usar o método `LateUpdate()` em vez do `FixedUpdate()`, e há dois bons motivos para essa escolha:

1. **Sincronização com a câmera**
   O `LateUpdate()` é chamado depois do `Update()`, o que garante que o cenário se mova somente após a posição da câmera ser atualizada. Isso evita aquele efeito de "tremor" na imagem que acontece quando a câmera e o cenário são atualizados em momentos diferentes.
2. **Eficiência**
   O `FixedUpdate()` é voltado para cálculos físicos e roda em intervalos fixos (por exemplo, 50 vezes por segundo), enquanto o `LateUpdate()` acompanha a taxa de quadros do jogo, tornando a sincronização mais precisa e suave.

5.1.3 Problemas em jogos grandes

Em projetos maiores, a escolha errada entre `FixedUpdate()` e `LateUpdate()` pode gerar alguns desafios:

- **Desempenho:** Se o jogo tiver muitas camadas de parallax, o `FixedUpdate()` pode acabar executando cálculos desnecessários, sobrecarregando o desempenho.
- **Inconsistência visual:** Como o `FixedUpdate()` roda em uma frequência fixa e a câmera é atualizada no `Update()`, isso pode causar pequenas diferenças de tempo entre os movimentos, gerando travamentos ou falhas no efeito de parallax.

Por isso, ao trabalhar com movimentação baseada na câmera, o `LateUpdate()` costuma ser a melhor escolha para garantir uma experiência mais fluida e visualmente agradável.

5.2 Collision Detection

A detecção de colisões foi implementada utilizando o sistema de física 2D da Unity. O método **Physics2D.OverlapBox** verifica se Bento está em contato com o solo antes de permitir um salto. Esse sistema:

- **Evita saltos fantasmas**, garantindo que o personagem só pule quando estiver sobre uma superfície válida.
- **Otimiza a performance**, filtrando colisões desnecessárias por meio do **LayerMask**.
- **Melhora o feedback visual**, permitindo ajustes precisos no desenvolvimento.

Essa abordagem proporciona uma jogabilidade responsiva e equilibrada, contribuindo para a experiência do jogador.

5.3 Gravity Manipulation

Para aprimorar a dinâmica dos saltos e quedas, desenvolvemos um sistema de gravidade personalizado. Diferente do padrão da Unity, ele adapta a gravidade de acordo com a situação:

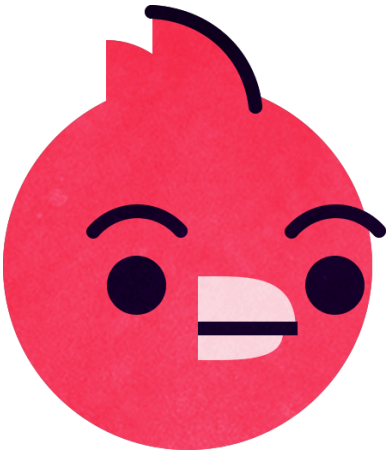
- **Queda Acelerada:** Quando Bento está em queda, a gravidade é aumentada para evitar uma sensação de "flutuação".
- **Ascensão Controlada:** Durante a subida, a gravidade é reduzida para um salto mais suave.

Isso permite que o jogador controle a altura dos pulos ao segurar ou soltar o botão de salto, adicionando profundidade à mecânica.

6. ASSETS

6.1 PERSONAGEM

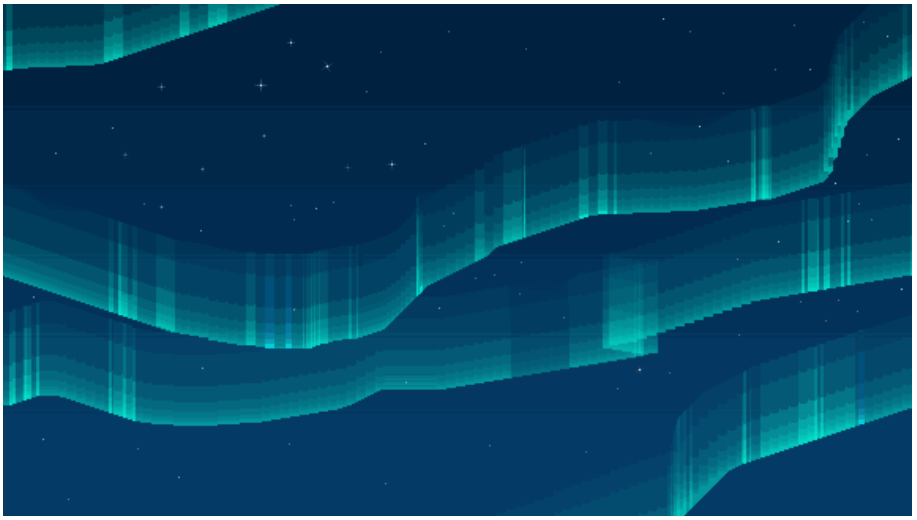
Figura 1 - Pássaro



Fonte: AUTOR DESCONHECIDO. Pássaro, [s.d.]. Disponível em: https://www.dropbox.com/...c&dl=0. Acesso em: 10 Março. 2025.

6.2 CENÁRIO

Figura 2 - Aurora boreal



Fonte: FREE GAME ASSETS. Nature Landscapes – Free Pixel Art, [s.d.]. Disponível em: https://free-game-assets.itch.io/nature-landscapes-free-pixel-art. Acesso em: 10 Março. 2025.

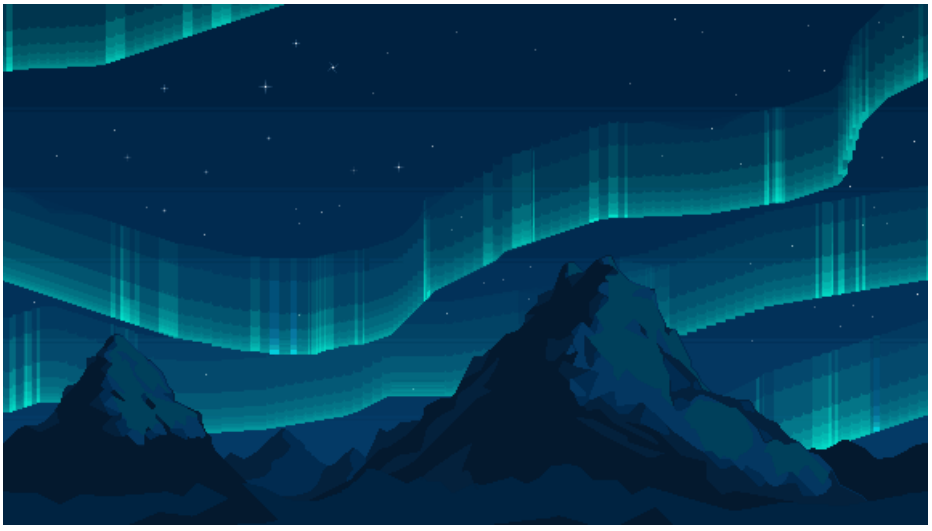
Figura 3 - Céu

Fonte: FREE GAME ASSETS. Nature Landscapes – Free Pixel Art, [s.d.]. Disponível em: https://free-game-assets.itch.io/nature-landscapes-free-pixel-art. Acesso em: 10 Março. 2025.

Figura 4 - Montanhas

Fonte: FREE GAME ASSETS. Nature Landscapes – Free Pixel Art, [s.d.]. Disponível em: https://free-game-assets.itch.io/nature-landscapes-free-pixel-art. Acesso em: 10 Março. 2025.

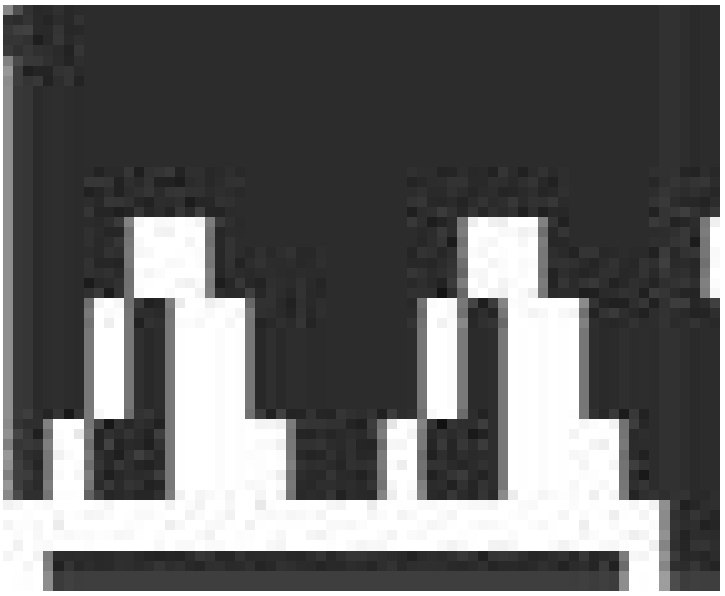
Figura 5 - Cenário Completo



Fonte: FREE GAME ASSETS. Nature Landscapes – Free Pixel Art, [s.d.]. Disponível em: https://free-game-assets.itch.io/nature-landscapes-free-pixel-art. Acesso em: 10 Março. 2025.

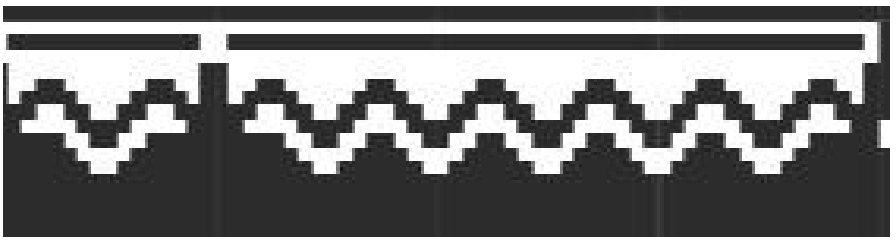
6.3 OBJETOS

Figura 6 - Espinhos



Fonte: KENNEY. 1-Bit Platformer Pack, [s.d.]. Disponível em: https://www.kenney.nl/assets/1-bit-platformer-pack. Acesso em: 10 Março. 2025. Licença: CC0 1.0 Universal.

Figura 7 - Plataformas



Fonte: KENNEY. 1-Bit Platformer Pack, [s.d.]. Disponível em: https://www.kenney.nl/assets/1-bit-platformer-pack. Acesso em: 10 Março. 2025. Licença: CC0 1.0 Universal.

7.CONCLUSÕES

Desenvolver este projeto foi uma experiência enriquecedora, permitindo que colocássemos em prática diversas técnicas de computação gráfica aprendidas ao longo do semestre. Foi gratificante ver o jogo ganhando forma e se tornando uma experiência jogável e visualmente envolvente. A implementação do Parallax Scrolling ajudou a criar uma sensação de profundidade no cenário, tornando o ambiente mais dinâmico. No entanto, encontramos alguns desafios pelo caminho, como a tela piscando em certos momentos devido a falhas na sincronização do efeito de parallax.

Além disso, a manipulação personalizada da gravidade e a detecção de colisões trouxeram uma jogabilidade mais responsiva, mas ainda há espaço para refinamentos, especialmente para evitar situações inesperadas, como saltos imprecisos ou quedas abruptas.

No futuro, pretendemos fazer melhorias para otimizar a performance, aprimorar os efeitos visuais e corrigir pequenos bugs que foram identificados durante os testes. Com essas melhorias, o jogo poderá oferecer uma experiência ainda mais fluida e envolvente, tornando-se um projeto ainda mais sólido e bem polido.