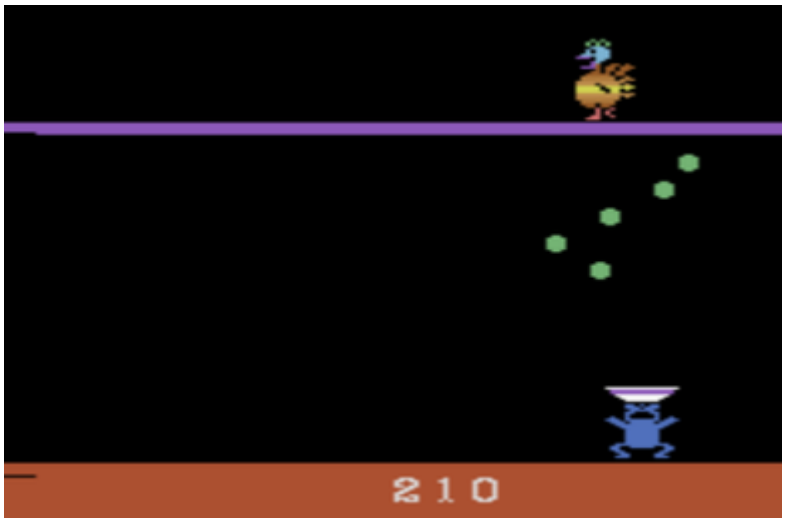
UNISINOS

Trabalho Prático – Fundamentos de Computação Gráfica

Desenvolvimento de um Jogo 2D com Sprites



Exemplo de gameplay do jogo Eggomania para Atari (1983)

Individual ou grupos de até 3 participantes

DATA DE ENTREGA: até 07/12/2024, via Moodle
Instruções para envio do trabalho: Apenas 1 integrante do grupo deve enviar o link para o repositório do projeto na atividade aberta no Moodle até as 23h59min do dia 07/12/2024. O diretório do projeto deve conter:
- O código fonte do trabalho.
- Um arquivo `LEIAME.md` com o nome completo dos integrantes do grupo e instruções de uso do programa.
- Exemplos de entrada e saída esperados para facilitar a correção.

Introdução

O jogo **Catch**, desenvolvido pela Atari em 1977, foi um protótipo de um jogo baseado em bola e paddle. Apesar de nunca ter sido lançado, ele inspirou diversos jogos futuros. Catch utilizava gráficos em preto e branco e foi cancelado, mas sua ideia foi reutilizada em **Avalanche**, lançado em 1978 com gráficos coloridos. Avalanche trouxe como inovação a temática de rochas caindo e foi bem-recebido pelo público. Curiosamente, o manual do Atari informa que Avalanche começou como um jogo chamado **Catch**, onde o jogador deveria coletar ovos em cestas, mas a temática foi alterada para pedras e tornou-se um grande sucesso.

Outros jogos que marcaram época e trouxeram variações dessa mecânica foram **Eggomania** (1982), em que o jogador utiliza um chapéu para pegar ovos lançados por uma galinha, e **Kaboom!** (1981), que adicionou mais intensidade ao gênero ao exigir que o jogador capturasse bombas rapidamente.

Jogos com mecânicas de “catchers” são aqueles em que o jogador controla um elemento móvel, geralmente na parte inferior da tela, com o objetivo de interceptar ou coletar objetos que caem de cima. Essa mecânica exige reflexos rápidos e precisão, proporcionando uma experiência de jogo desafiadora e envolvente.

Características principais desses jogos

- **Movimento Horizontal**: O jogador pode mover o elemento coletor (como uma cesta, plataforma ou personagem) horizontalmente para posicioná-lo adequadamente sob os objetos que caem.
- **Objetos em Queda**: Itens caem de forma aleatória ou em padrões específicos da parte superior da tela, e o jogador deve interceptá-los antes que atinjam o chão ou saiam da tela.
- **Aumento de Dificuldade**: Conforme o jogo avança, a velocidade e a frequência dos objetos em queda geralmente aumentam, tornando o desafio mais intenso.

Essas mecânicas têm sido amplamente utilizadas para criar jogos simples, intuitivos e altamente engajantes, sendo uma ótima base para desenvolvimento de habilidades de programação e design de jogos.

Objetivo

O objetivo deste trabalho é desenvolver um protótipo de jogo 2D, utilizando a API Gráfica OpenGL (moderna), aplicando os conceitos vistos em aula: desenho de primitivas gráficas, transformações geométricas, projeção ortográfica e mapeamento de textura serão explorados.

Para isso, sugere-se desenvolver um jogo com a mecânica de catch. O jogo deverá utilizar a lógica de sprites para representar os elementos da cena, permitir o controle/movimentação de um personagem e incluir animações básicas baseadas em spritesheets em pelo menos um objeto (como personagem ou itens que caem).

Descrição do Problema

O jogo deverá conter:

1. **Movimentação do Personagem**
    - Controle do personagem usando as setas do teclado ou o mouse para se deslocar (movimento horizontal, mas pode adaptar para outras direções se preferir).
2. **Itens Coletáveis e/ou para desviar**
    - Itens caindo aleatoriamente com velocidades variadas.
    - Detecção de colisão entre o personagem e os itens.
    - Se for um coletável, aumenta pontuação
    - Se for um item para se desviar, calcular o dano.
3. **Animações**

   - Utilização de spritesheets para animar o personagem principal durante o movimento.
4. **Pontuação e Dano - Colisões**
   - Cada item coletado incrementa a pontuação do jogador.
   - Se houverem itens que precisam ser evitados e o personagem colidir com eles, deve-se causar algum tipo de dano. Por exemplo: o personagem tem 3 vidas e desconta uma delas, ou vai baixando o HP até ele "morrer".
   - Implementar uma contagem para o jogador acompanhar seu desempenho. Como implementar textos diretamente com OpenGL é uma tarefa trabalhosa, o feedback dos pontos pode ser dado pelo terminal.
   - Implementar uma estratégia de Game Over (baseado no nro de pontos ou até o personagem "morrer").

Requisitos Técnicos

1. Utilizar a **API OpenGL na versão 3.3 ou superior**, o que caracteriza que é obrigatória a implementação de **shaders**.
2. Correta criação e utilização correta dos buffers de geometria
   - VAO, VBO(s) e EBO (este último é opcional)
3. Correta utilização das transformações de projeção e nos objetos
   - Como está mapeada a matriz de projeção ortográfica 2D?
   - Como são feitas e atualizadas as transformações nos objetos da cena?
4. **Sprites** como textura mapeadas em polígonos, para o desenho de:
   - personagem, objetos e itens animados (utilizando spritesheets)
   - imagens de fundo (uso de camadas)
     - em fundos multi-camadas podemos implementar o conceito de **parallax scrolling**
5. O spawn dos itens deve ser feita de maneira aleatória, preferencialmente seguindo uma direção (como de cima para baixo).
6. Colisão entre os sprites (conceito de **hitboxes**), para ações como coletar itens, tiros etc., conforme a proposta do jogo)
7. O código deve ser organizado, modularizado e devidamente comentado.

Material Auxiliar

Os seguintes materiais estão disponíveis para auxiliar na implementação:

- Introdução aos Sprites
- Animação de Sprites

Referências

1. Schell, J. *The Art of Game Design: A Book of Lenses, Third Edition*. CRC Press, 2019.
2. Gamedesignskills.com - Game Mechanics

3. Wikipedia - Avalanche
4. Wikipedia - Eggomania
5. Arcade Museum - Catch
6. Arcade Museum - Avalanche
7. Learn OpenGL - Rendering Srpites

Bom trabalho! 😊

Desafios como este permitem que você transforme ideias em realidade. Explore sua criatividade, aplique o que aprendeu e divirta-se!