Visualização e Animação 2D em OpenGL com Dados Reais do Cultural Dataset

Giovani Cancherini[1]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Porto Alegre – RS – Brasil

giovani.cancherini@edu.pucrs.br

Abstract. *This paper presents the development of a 2D interactive application using OpenGL, based on real human movement data extracted from the Cultural Dataset. The application simulates an environment where the player must avoid animated entities, whose trajectories are based on this data. At each time interval, the difficulty increases. The project focuses on graphical visualization, event handling, collision detection, and real-time rendering. Visual results, the pipeline, and future improvement perspectives are discussed.*

Resumo. *Este artigo apresenta o desenvolvimento de uma aplicação interativa 2D com OpenGL, baseada em dados reais de movimentação humana extraídos do Cultural Dataset. A aplicação simula um ambiente onde o jogador deve desviar de entidades animadas, cuja trajetória é baseada nesses dados. A cada intervalo de tempo, a dificuldade aumenta. O projeto foca em visualização gráfica, controle de eventos, colisões e renderização em tempo real. Resultados visuais, pipeline e perspectivas de melhorias são discutidos.*

1. Introdução e Contextualização

Este trabalho apresenta o desenvolvimento de uma aplicação de visualização e animação 2D utilizando a biblioteca OpenGL, no contexto da disciplina de Computação Gráfica. A proposta central foi criar uma experiência interativa com aspecto lúdico, aproximando-se de um jogo. O usuário controla um triângulo branco em um ambiente bidimensional e deve evitar colisões com entidades representadas por quadrados que percorrem trajetórias reais extraídas do arquivo "Paths_D.txt", parte do "Cultural Dataset". Essas trajetórias são formações de coordenadas associadas a quadros de um vídeo em que pessoas caminham, fornecendo dados ricos e reais para simulação.

2. Objetivo do Projeto

O objetivo geral do projeto foi explorar os fundamentos da renderização gráfica e da manipulação de animações em duas dimensões com OpenGL, utilizando dados reais como base para a simulação. Mais especificamente, buscou-se implementar uma aplicação interativa que incorporasse elementos de jogabilidade, controle por teclado, resposta visual e incremento de dificuldade, a partir de dados oriundos de trajetórias reais. A escolha do formato gamificado teve como motivação principal favorecer o engajamento do usuário durante a interação, ao mesmo tempo em que se aplicavam conceitos fundamentais da computação gráfica. Parte das decisões iniciais de estruturação do projeto e uso das bibliotecas foram guiadas por exemplos visuais disponíveis em tutoriais da série *OpenGL with Python*, do canal Get Into Game Dev [Get Into Game Dev 2020].

3. Recursos do OpenGL Utilizados

A aplicação utiliza uma variedade de recursos oferecidos pela OpenGL. Entre eles, destacam-se o uso de primitivas geométricas como `GL_TRIANGLES` e `GL_QUADS`, responsáveis por desenhar o triângulo controlado e as entidades móveis. Esses conceitos foram reforçados com base em materiais clássicos da literatura como Cohen e Manssour [Cohen and Manssour 2006], que abordam OpenGL de maneira prática e didática. A manipulação de cores RGB foi empregada para diferenciar as entidades na cena, com uma paleta específica e utilizando do número de frames de cada entidade como identificador, sendo usado um algoritmo derivado para geração de cores únicas disponíveis publicamente na comunidade de desenvolvedores [Stack Overflow 2009]. Enquanto as funções de transformação de matriz, como `glTranslatef`, `glPushMatrix` e `glPopMatrix`, permitiram o controle da posição dos objetos e da câmera. A biblioteca GLUT foi essencial para a criação da janela, configuração da projeção ortográfica por meio de `gluOrtho2D`, e tratamento de eventos de entrada (teclado), bem como para o controle da função de desenho contínuo via `glutIdleFunc`.

4. Execução e Avaliação dos Resultados

Para executar a aplicação, é necessário instalar previamente as dependências listadas no arquivo `requirements.txt`, utilizando o comando `pip install -r requirements.txt`. Com os requisitos instalados, a aplicação pode ser executada diretamente via `python t1.py`. Durante a execução, um arquivo de dados é selecionado aleatoriamente da pasta `Brazil - BR`, sendo carregado em tempo real e utilizado como fonte das animações das entidades.

Os resultados visuais da aplicação são observáveis diretamente na janela gráfica gerada pelo OpenGL. As entidades se movimentam conforme suas trajetórias temporais, enquanto o triângulo controlável reage em tempo real aos comandos do jogador. A dificuldade aumenta dinamicamente, à medida que o tempo avança, tornando a animação mais veloz. Quando ocorre uma colisão, o jogo é pausado e um som é emitido, reforçando a percepção de falha. O sistema de pontuação é atualizado com base no tempo de sobrevivência do jogador, e novas mensagens são impressas no console para indicar o progresso.

5. Uso de Conceitos de Inteligência Artificial

Embora o projeto não tenha adotado técnicas tradicionais de inteligência artificial, como redes neurais ou sistemas especialistas, foram aplicados conceitos de apoio oriundos de ferramentas de IA generativa. A ferramenta ChatGPT foi utilizada como meio para explorar soluções viáveis para o tratamento de eventos com múltiplas teclas pressionadas, possibilitando a movimentação combinada do triângulo em diagonais e o controle da câmera com teclas modificadoras como CTRL. Além disso, foi aplicada uma solução para leitura dinâmica dos arquivos da pasta de dados, permitindo aleatorizar a simulação em cada nova execução, favorecendo a variedade de cenários.

6. Pipeline do Método Desenvolvido

A estrutura do projeto foi concebida com base em um pipeline lógico que engloba as seguintes etapas: (i) inicialização e carregamento dos dados, incluindo a seleção aleatória de um arquivo válido e leitura de coordenadas de movimentação; (ii) cálculo dos limites extremos de coordenadas (mínimos e máximos), a fim de configurar a área visível da cena com margem apropriada; (iii) renderização das entidades e do triângulo

com primitivas geométricas coloridas; (iv) controle de interação com o jogador por meio de eventos de teclado, permitindo movimentação em tempo real; (v) atualização de estado das entidades com base em tempo contínuo (frames); e (vi) verificação de colisões utilizando bounding boxes, além do controle de pontuação e mensagens no terminal. O uso de variáveis globais foi uma estratégia adotada para persistência de estado entre chamadas de funções e quadros de renderização, simplificando o compartilhamento de parâmetros centrais do jogo.

7. Melhorias Futuras

Dentre os pontos de aprimoramento identificados, destaca-se a necessidade de substituir o algoritmo atual de colisão por uma abordagem mais precisa. Embora bounding boxes sejam simples e eficientes, elas não cobrem adequadamente a geometria de entidades não retangulares. Uma possível solução seria a implementação de testes de interseção poligonal por meio do Separating Axis Theorem (SAT). Além disso, pretende-se incluir entidades com trajetórias geradas aleatoriamente, independentemente dos dados reais, de modo a aumentar o desafio do jogo conforme o tempo avança. Essa sugestão foi feita pela professora da disciplina e visa intensificar o dinamismo da simulação. Por fim, a inclusão de uma interface gráfica mais informativa dentro da própria janela OpenGL é desejável, substituindo a atual saída de dados no terminal.

8. Conclusão

Conclui-se que o projeto atingiu plenamente os objetivos propostos, utilizando com eficiência os recursos da biblioteca OpenGL e proporcionando uma experiência interativa baseada em dados reais. O emprego de IA no planejamento e solução de pequenos desafios de código complementa o desenvolvimento autoral e estruturado. A aplicação demonstrou estabilidade, clareza de propósito e potencial para expansão futura com novas camadas de complexidade e interatividade.

9. Imagens

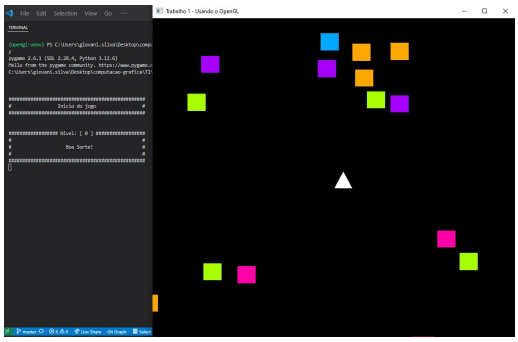

Figura 1. Exemplo da janela principal da aplicação durante execução.

Referências

Cohen, M. and Manssour, I. (2006). *OpenGL - Uma Abordagem Prática e Objetiva*. Novatec, São Paulo.

Get Into Game Dev (2020). Opengl with python - série de vídeos no youtube. Canal do YouTube. Disponível em: https://www.youtube.com/watch?v=LCK1qdp_HhQ\&list=PLn3eTxaOtL2PDnEVNwOgZFm5xYPr4dUoR.

Stack Overflow (2009). Algorithm for generating unique colors. Online. Disponível em: https://stackoverflow.com/questions/1168260/algorithm-for-generating-unique-colors.