Entregável 2 Computação Gráfica

Integrantes:

- Eduardo Garcia Fensterseifer
- Guilherme Beno kemmer Dalmoro
- Livia Souza da Silva

Repositório:

https://github.com/EduardoFen12/Entregavel2-cg.git

Elevator Pitch:

(Convolução): O Entregável 2, Convolução, é uma aplicação de Computação Gráfica que demonstra o poder do processamento paralelo de imagens utilizando a pipeline programável da GPU (OpenGL/GLSL). Ele permite aplicar filtros de convolução 3x3 (como Blur, Sharpen, Emboss e Detecção de Bordas) em uma textura 3D em tempo real. O sistema é controlado pelo teclado, permitindo a alternância instantânea entre diferentes kernels para uma visualização imediata e eficiente do efeito na imagem.

O Entregável 2, Manipulação de Câmera em 3D, é uma aplicação de Computação Gráfica que demonstra o poder do processamento paralelo de imagens utilizando a pipeline programável da GPU (OpenGL/GLSL). Um cubo 3D é controlado pelo teclado e mouse, permitindo a navegação no espaço 3D (matrizes MVP).

Como Rodar o Projeto:

Para executar a aplicação, você precisará de uma máquina com suporte a OpenGL 3.3+ e as seguintes dependências em Python. Esses passos servem tanto para a atividade de Convolução, quanto para a de 3D.

Requisitos

O projeto utiliza bibliotecas comuns para gráficos e visão computacional, como:

- numpy
- PyOpenGL (módulos OpenGL.GL e ctypes)
- glfw
- Pillow (PIL.Image)
- opencv-python (cv2) (Opcional: usado apenas para o arquivo de comparação conv_opencv_G2.py)

> pip install numpy PyOpenGL glfw Pillow

Instruções de Execução

1. Organização: Certifique-se de que todos os arquivos do repositório (main.py, Fragment_shader.glsl, vertex_shader.glsl, fps_counter.py e a imagem templo.jpg) estão no mesmo diretório.
2. Execução: Execute o arquivo principal a partir do seu terminal:
> python main.py

Descrição das Funcionalidades

O projeto combina navegação 3D com a aplicação de filtros de imagem via Fragment Shader (convolução).

Processamento de Imagem (Filtros GLSL)

Os filtros de imagem são aplicados em tempo real no fragment shader, garantindo alto desempenho.

1. Convolução 3x3: O shader utiliza uma matriz uniforme (kernel[9]) para aplicar diversos efeitos de filtro 3x3.
    Controle: Alternados pelas teclas numéricas 1 a 5.

2. Modo Tons de Cinza (Gray Scale): Aplica a conversão de luminância padrão (Y = 0.299R + 0.587G + 0.114B) após a convolução.
    Controle: Ativado/desativado com o Botão Esquerdo do Mouse.

3. Reset: O kernel é restaurado para a matriz Identidade, mostrando a imagem original.
    Controle: Tecla 0 ou Botão Direito do Mouse.

Controles de Navegação e Câmera

O usuário pode interagir com o objeto 3D e a textura através dos seguintes comandos de câmera:

1. Movimento: Permite translação da câmera nos três eixos (X, Y e Z).
    Controle: W/S (Frente/Trás), A/D (Lado a Lado), SPACE/LEFT_SHIFT (Cima/Baixo).

2. Rotação: Permite rotação da câmera nos eixos X e Y.
    Controle: Setas direcionais (UP/DOWN/LEFT/RIGHT).

3. Reset de Posição: Restaura a câmera para a posição inicial (Z=5).
    Controle: Botão Direito do Mouse.

Aferição de Desempenho

1. Contador de FPS: Exibe o Frame Rate atual na tela.
2. Medição de Latência: Imprime no console o tempo de renderização da GPU sempre que um novo kernel é aplicado.

Melhorias Futuras (Roadmap)

1. Adicionar mais opções de filtros (kernels), como Inversão de Cores, Vignette ou Bloom
2. Melhoria na Exibição de FPS: Utilizar um método de renderização de texto mais eficiente, como o Freetype ou Text Rendering com Texturas, em vez de redesenhar vértices por caractere.
3. Gerenciamento de Shaders: Implementar uma classe para compilação e gerenciamento de shaders para simplificar a adição de novos programas Shader.