UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

INSTITUTO DE INFORMÁTICA

INF01009 / CMP143 Computação Gráfica

‘

Atividade 3

Inatan Lopes Hertzog 219420

Implementação

Nessa Atividade será dada continuidade na implementação da Engine Close2GL. Na última versão faltou o estágio de rasterização do pipeline gráfico Gouraud com z-buffer. Nessa etapa será implementado o z-buffer, rasterização, iluminação e shading. Posteriormente será implementado o mapeamento de texturas.

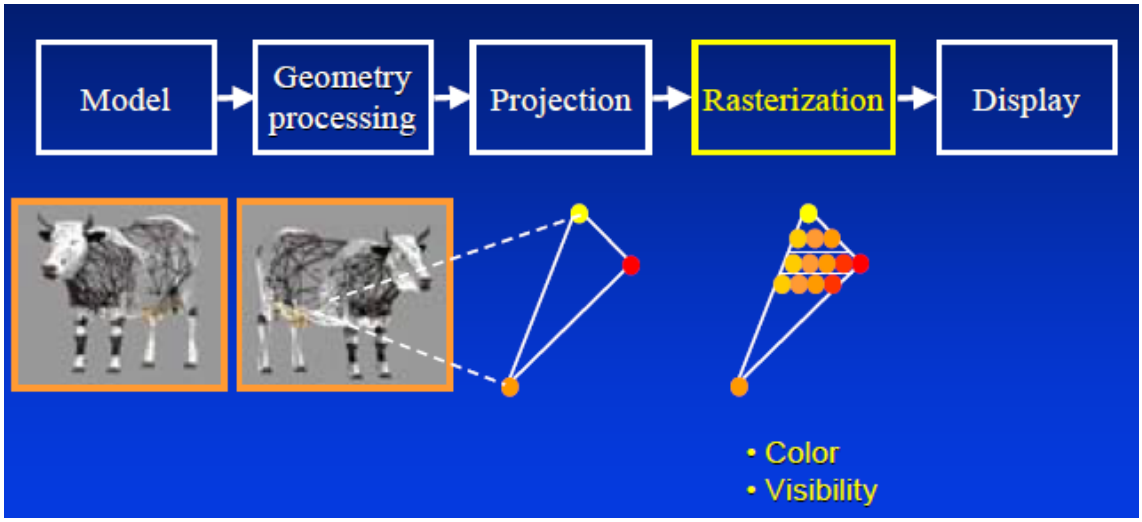

Figura 1 – *Momento atual*

Foram implementado rasterização em pontos, linhas e solido. No caso dos pontos foram só desenhados na tela, já para a rasterização de linha cada aresta foi separada e desenhada cada pixel do início ao fim. Por fim a rasterização de sólido é inicia com uma interpolação entre as arestas iniciando do x e y menores, após isso é dividido por w e gravado no buffer.

Na iluminação foram utilizados a formula de Phong. Onde:

- *Ka* constante de luz ambiente
- *Kd* é uma constante difusa do material
- *Ks* é uma constante especular do material
- *Fatt* fator de atenuação
- *Ia* = Componente ambiente
- *Id* = Componente difusa
- *Is* = Componente especular
- *N* sendo o vetor normal no ponto em questão da superfície
- *L* o vetor de incidência da luz
- *R* o vetor de reflexão da luz
- *V* o vetor de visão da camera
- *n* é intensidade especular

$$I = I_a k_a + f_{att} I_l \left(k_d (N \cdot L) + k_s (V \cdot R)^n \right)$$

Figura 2 – *modelo de iluminação de Phong*

Foram implementados os modelos de sombreamento com relação a fonte de luz.

- *Flat Shading*: Realiza o calculo de iluminação baseado nas normais dos objetos.
- *Gouraud Shading*: Obtém as normais dos vértices.

Por fim a última parte implementada seria a visibilidade através do *zbuffer.* O qual é um buffer do tamanho da janela que verifica a profundidade do pixel e aquele que estiver mais afrente é desenhado.

Projeto

O projeto onde foi desenvolvido foi o mesmo da atividade 2 com algumas mudanças de classes. Foram adicionadas mais classes voltadas ao projeto.

Em busca de um melhor desenvolvimento foram adicionadas classes para melhor organização de código.

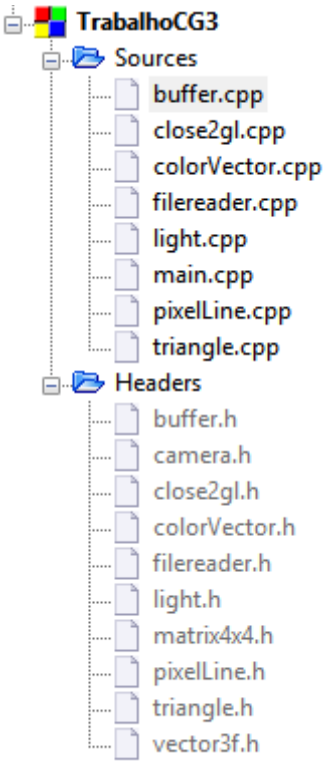

Figura 3 – *Mapa de classes do projeto.*

- **camera:** camera a qual é realizada as operações
- **close2gl:** funções principais da close2gl
- **filereader:** leitor de objeto

- **triangle:** representação de um triangulo
- **matrix4x4:** matrix utilizada para todos os cálculos no close2gl contem multiplicação de matrizes e multiplica uma matriz por um vetor de dimensão 4.
- **vector3f:** vetor para determinar posições de vértices utilizando agora w para coordenadas homogêneas
- **buffer:** buffer de profundidade e cores o tamanho máximo é o tamanho da janela
- **colorVector:** Voltado a simular um vetor de cores com R,G,B e A.
- **Light:** Tem a propriedades da luz que serão utilizados para o calculo de iluminação de Phong.
- **pixelLine:** linhas para desenhar arestas ou rasterizar pixe contendo dois pontos, mais suas profundidades e a cor de cada.
- **main:** contem a interface e gerencia todas as classes

A interface de controle foi alterada agora o campo de luz permite alterar, as constantes ambiente, difusa e especular. Foram adincinados campos permitindo o tipo de shading. E por fim um checkbox ativando o zbuffer.

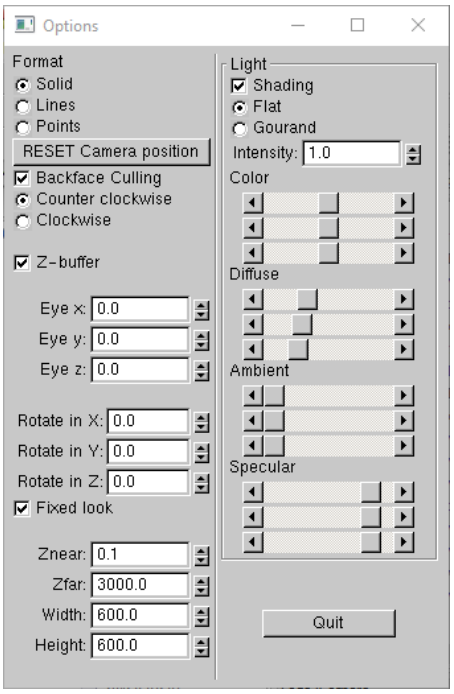

Figura 3 – *Nova interface de opções*

Resultados

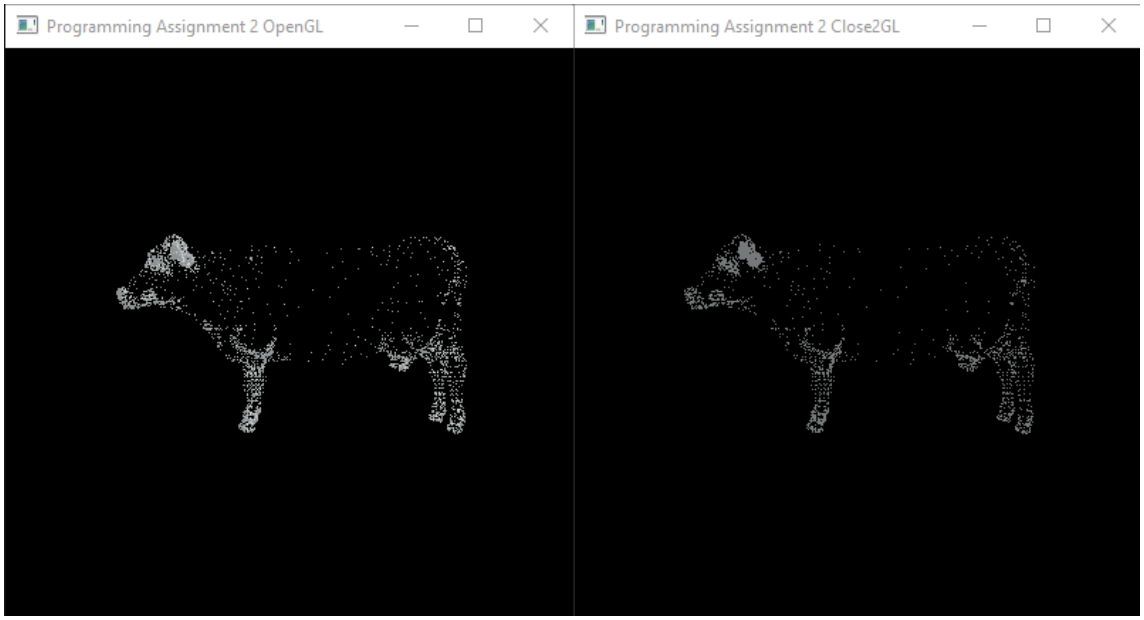

Figura 4 – *Objeto em points*

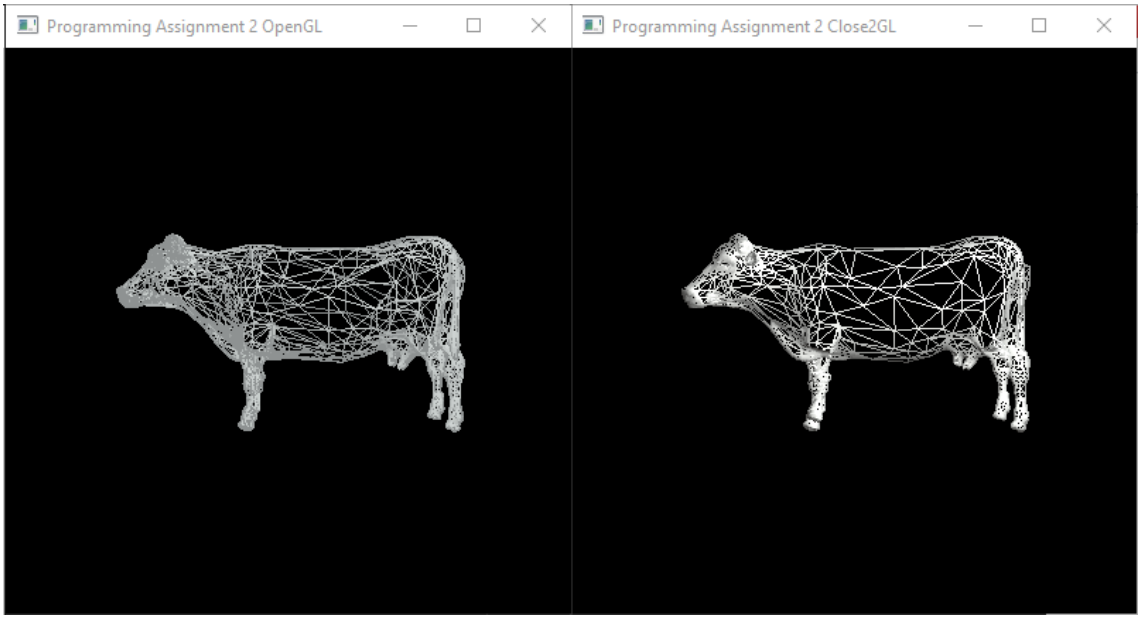

Figura 5 – *Objeto em wireframe*

Figura 6 – *Objeto em solid*

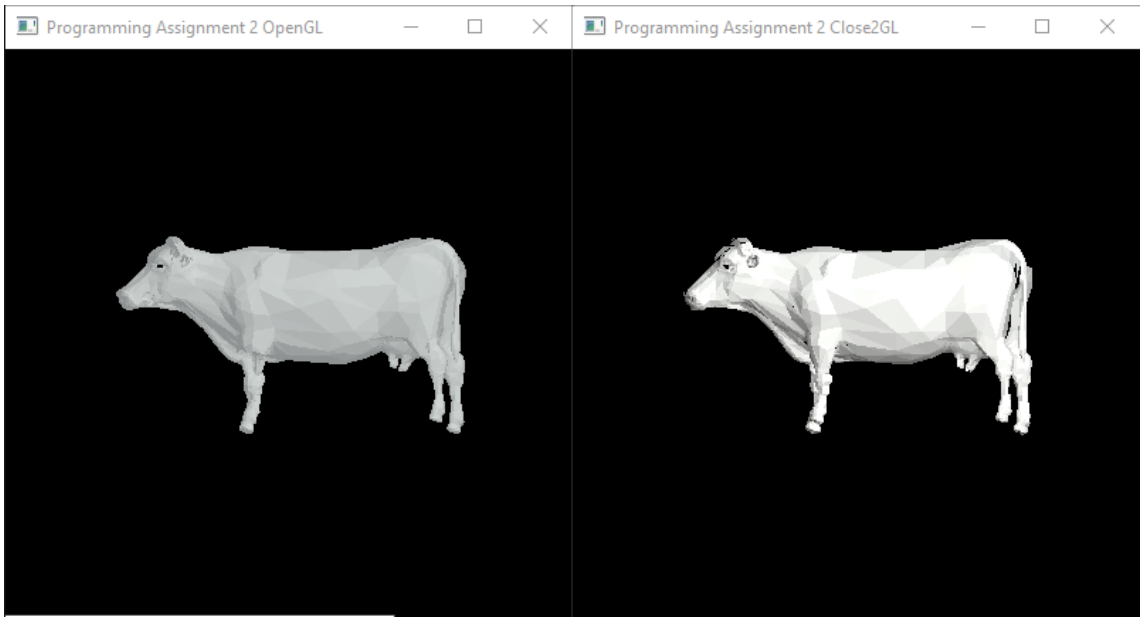

Figura 7 – *Objeto em flat shading*

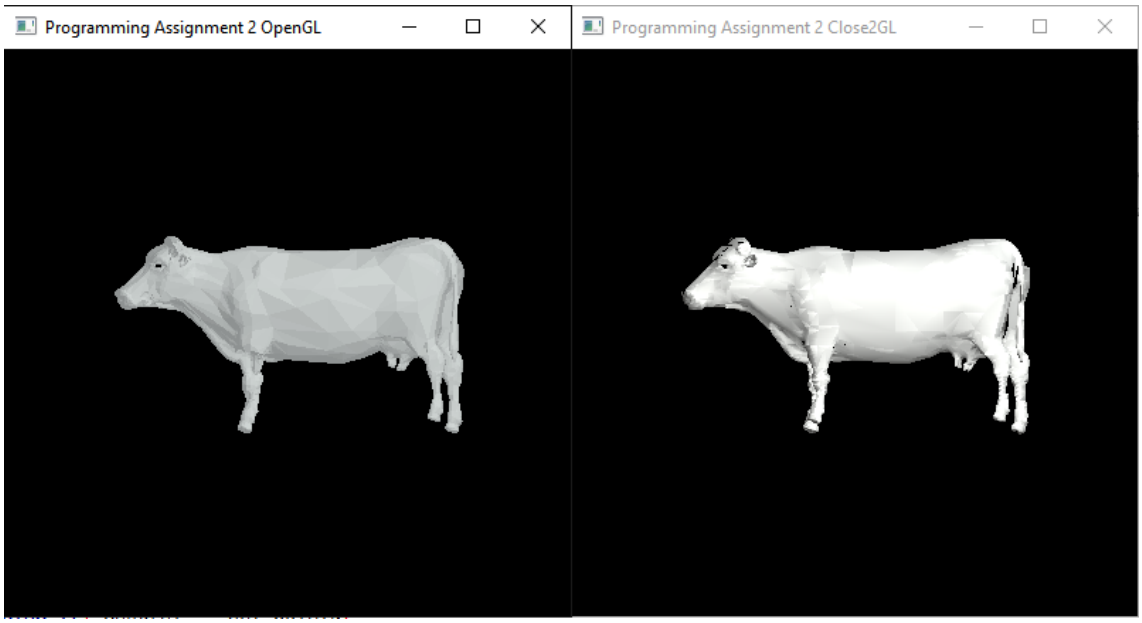

Figura 8 – *Objeto close2gl em Gouraud*

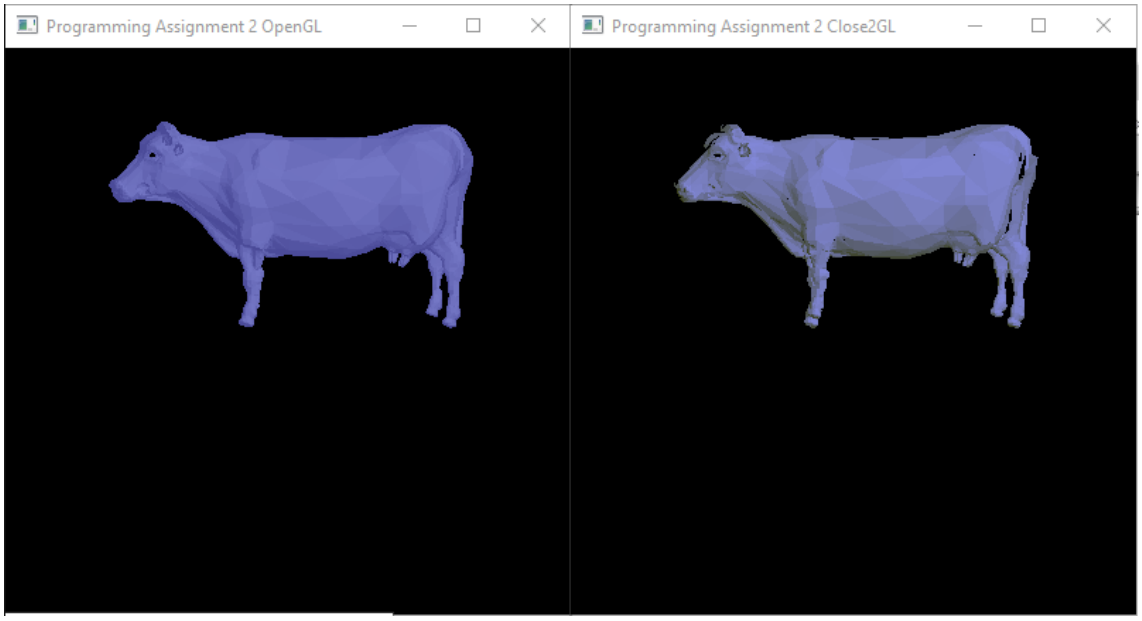

Figura 10 – *Zbuffer desativado pode notar que o rabo e a perna da jysel estão sobre postos nas duas janelas (zfighting).*

Conclusão

Nessa etapa foi implementado a parte de rasterização com visibilidade. Foi entendido melhor o funcionamento da mecânica da luz e como pode-se presentar, a sua interação com o objeto(shading) e a importância do zbuffer para evitar qualquer problema de zfighting.