PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
DEPARTAMENTO DE COMPUTAÇÃO
COMPUTAÇÃO GRÁFICA CMP 1170 – 2019/1
PROF. MSC. GUSTAVO VINHAL

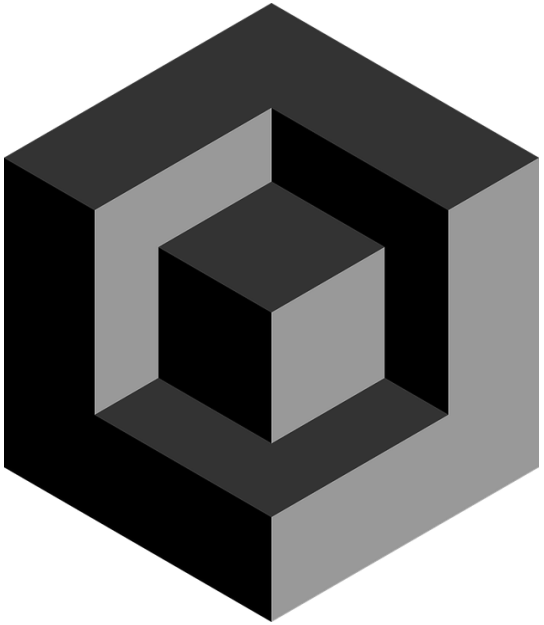
Aula 13
Realismo Visual e Iluminação (Continuação)

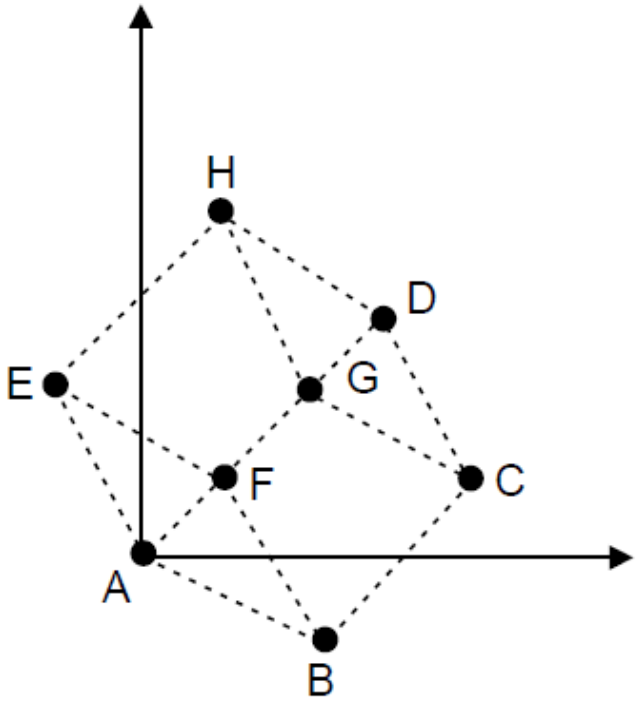
PUC GOIÁS

Rasterização

Eliminação de faces escondidas (*culling back-faces*)

- Devido a posição do objeto em relação ao observador, algumas faces estarão ocultas;
- Essa etapa remove as faces.





PUC GOIÁS

Rasterização

Eliminação de faces escondidas (*culling back-faces*)

- O uso apropriado de técnicas de projeção e eliminação de superfícies (e linhas) não visíveis, auxilia em criar resultados mais realistas.

- Para exibição de objetos complexos, pode-se utilizar duas abordagens:

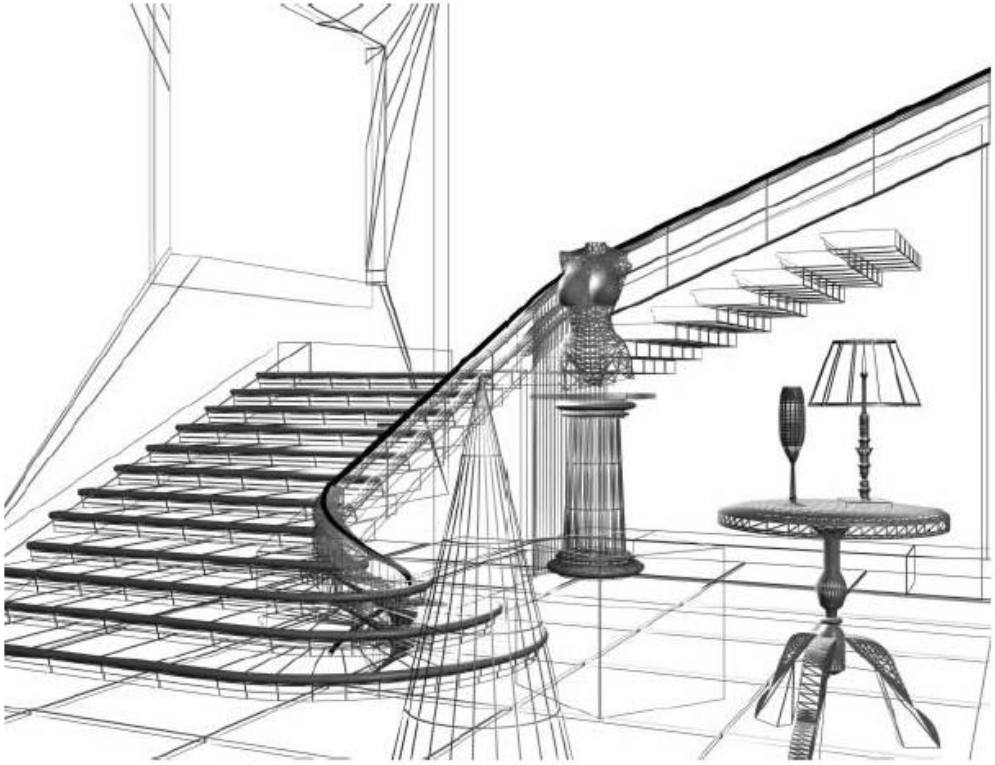
  - Exibição por um subconjunto de retas e curvas;
  - Exibição por um subconjunto de faces.

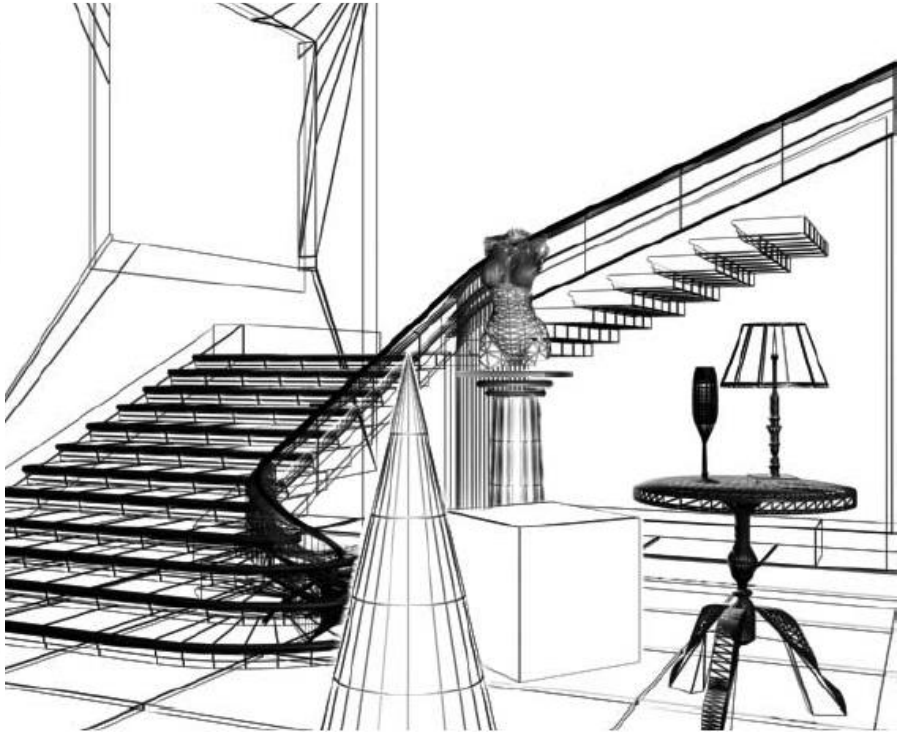
PUC GOIÁS

Rasterização

Eliminação de faces escondidas (*culling back-faces*)

- Exibição por um subconjunto de retas e curvas;
  - Objeto gerado processando suas linhas escondidas.





PUC GOIÁS

Rasterização

Eliminação de faces escondidas (*culling back-faces*)

- Exibição por um subconjunto de faces (mais utilizado)
    - Objeto gerado processando suas faces escondidas.
- Algoritmos específicos para realizar essa etapa (HSR – *Hidden Surface Removal*)
    - Problemas: consumo excessivo de memória ou tempo de computação.
        - Memória maior e mais barata = implementação de algoritmos com consumo excessivo de memória.
- Algoritmos mais utilizados:
    - Algoritmo de Visibilidade por Prioridade;
    - Algoritmo de Eliminação de Faces Ocultas pelo Cálculo da Normal;
    - Algoritmo Z-Buffer.

PUC GOIÁS

Rasterização - Eliminação de faces escondidas

Algoritmo de Visibilidade por Prioridade (algoritmo do pintor)

- Ideia: *Se um objeto A bloqueia a visão de um objeto B e ambos os objetos estão na mesma linha de visão do observador, então o objeto B está mais distante do observador que o objeto A.*
- Ponto principal: distância.

$$D = \sqrt{x^2 + y^2 + z^2} = |x| + |y| + |z|$$

x, y, z = coordenadas do centroide da face

- Etapas:
  - Calcula-se a distância (D) ao observador de todas as faces poligonais da cena;
  - Ordenam-se todos os polígonos pelo valor da sua distância;
  - Resolvem-se as ambiguidades (se as distâncias são iguais, verificam-se se as posições são iguais);
  - Desenham-se primeiro os polígonos que tiverem mais distantes do observador (os objetos mais distantes são sobrescritos).

PUC GOIÁS

Rasterização

Rasterização de retas – Algoritmo de Eliminação de Faces Ocultas pelo Cálculo da Normal

- Outra característica das superfícies (que não a distância) é o ângulo que a sua normal faz com a direção do observador.

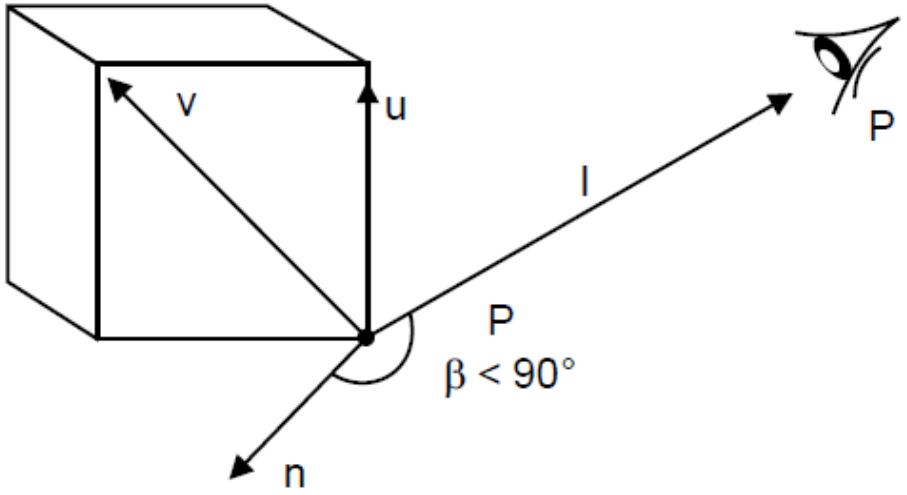
- Ponto principal: ângulo entre normal e o vetor de visibilidade do observador

- Etapas:

    - Ler as coordenadas tridimensionais do objeto e armazená-las em forma de matriz;
    - Localizar no espaço 3D a posição do observador;
    - Calcular o vetor normal 3D de cada face do objeto;
    - Calcular o vetor da linha de visibilidade para cada face do objeto;
    - Realizar o teste de visibilidade:
        - Se o ângulo (β) estiver entre 90º e 180º a superfície estará invisível;
        - Se β estiver entre -90º e 90º a superfície estará visível.

Rasterização

Rasterização de retas – Algoritmo de Eliminação de Faces Ocultas pelo Cálculo da Normal



- **u** e **v** são dois vetores (obtidos por três pontos quaisquer (ordem importa)) localizados na face (superfície) do objeto;
- **l** é o vetor da linha de visibilidade (que liga a face ao olho do observador);
- **n** é o vetor normal (produto vetorial entre **v** e **u**).

PUC GOIÁS

Rasterização

Rasterização de retas – Algoritmo Z-Buffer

- Algoritmo mais utilizado. Pode ser implementado via *hardware* e/ou *software*

- Ponto principal: utilizar dois *buffers* (matrizes em memória com mesma dimensão da tela de apresentação) – Imagem e Profundidade

    - *Buffer* de imagem: “rascunho” durante os cálculos de visibilidade;
    - *Buffer* de profundidade (z-buffer): armazena a distância entre cada pixel da tela de rascunho fictícia e o plano de projeção.

- Antes do cálculo e da projeção dos objetos, inicializa-se:

    - O *buffer* de profundidade com os mais altos valores possíveis (representando distância máxima);
    - O *buffer* de imagem com os valores das cores de fundo da imagem.

PUC GOIÁS

Rasterização

Rasterização de retas – Algoritmo Z-Buffer

- Etapas:
  - Criar e inicializar com a cor de fundo uma matriz, que conterá a informação de cada pixel da tela *(buffer* de imagem)*;*
  - Inicializar um matriz com o valor de máxima profundidade *(buffer* de projeção)*;*
  - Achar a coordenada z para cada ponto do polígono;
  - Testar a profundidade z de ponto de cada superfície para determinar a mais próxima do observador;
  - Atualizar o valor nas matrizes se z estiver mais próximo do observador.

```
Para cada polígono P da cena
 Para cada pixel (x, y) de um polígono P
   computar z_depth na posição x, y
   se z_depth < z_buffer (x, y) então
      defina_pixel (x, y, color)
      troque o valor : z_buffer (x, y) = z_depth
```

PUC GOIÁS

Culling Face em OpenGL

- Para ativar o *culling face*:

```
glEnable(GL_CULL_FACE);
```

- Uma vez ativado, o OpenGL deve-se decidir qual face ocultar, podendo utilizar uma das três opções:

```
glCullFace(GL_BACK);
glCullFace(GL_FRONT);
glCullFace(GL_FRONT_AND_BACK);
```

PUC GOIÁS

Z-Buffer em OpenGL

- O z-buffer vem desativado por padrão no OpenGL. Para ativá-lo utiliza o comando:

```
glEnable(GL_DEPTH_TEST);
```

- Uma vez ativado, o OpenGL armazena os fragmentos que passam no teste (padrão é o GL_LESS) no z-buffer e descarta o restante.

Function	Description
GL_ALWAYS	The depth test always passes.
GL_NEVER	The depth test never passes.
GL_LESS	Passes if the fragment's depth value is less than the stored depth value.
GL_EQUAL	Passes if the fragment's depth value is equal to the stored depth value.
GL_LEQUAL	Passes if the fragment's depth value is less than or equal to the stored depth value.
GL_GREATER	Passes if the fragment's depth value is greater than the stored depth value.
GL_NOTEQUAL	Passes if the fragment's depth value is not equal to the stored depth value.
GL_GEQUAL	Passes if the fragment's depth value is greater than or equal to the stored depth value.

Z-Buffer em OpenGL

- Antes de utilizar o z-buffer é sempre bom limpar o buffer com o comando:

```
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
```

- Exercício:

1. Modelar o sol e a terra, com seus respectivos movimentos.

PUC GOIÁS

REFERÊNCIAS BIBLIOGRÁFICAS:

AZEVEDO, Eduardo; CONCI, Aura. **Computação gráfica:** teoria e prática. Rio de Janeiro: Campus, 2003.