Computação Gráfica e Realidade Virtual

Iluminação em OpenGL

Prof. Dr. Fernando Kakugawa

fernando.kakugawa@animaeducacao.com.br

Universidade Anhembi Morumbi

Tópicos

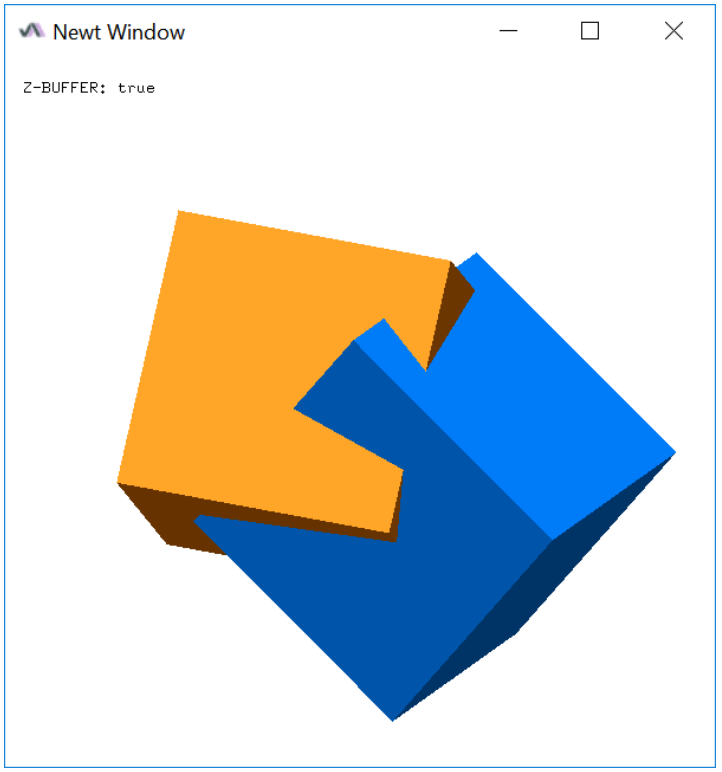
- Algoritmo z-buffer
- Modelos de Tonalização – Colorização (Shading)
- Iluminação e Tonalização em OpenGL

Algoritmo z-buffer

- Quando se tem cenários complexos é necessário um algoritmo de visualização para representar somente as partes visíveis da cena;
- O z-buffer é um algoritmo simples e implementado na placa de vídeo;
- Salva, em uma matriz 2D, o componente de profundidade de cada pixel;
- Determina quais faces da cada objeto que realmente devem aparecer na cena e quais não!

Algoritmo z-buffer

Pacote zbuffer



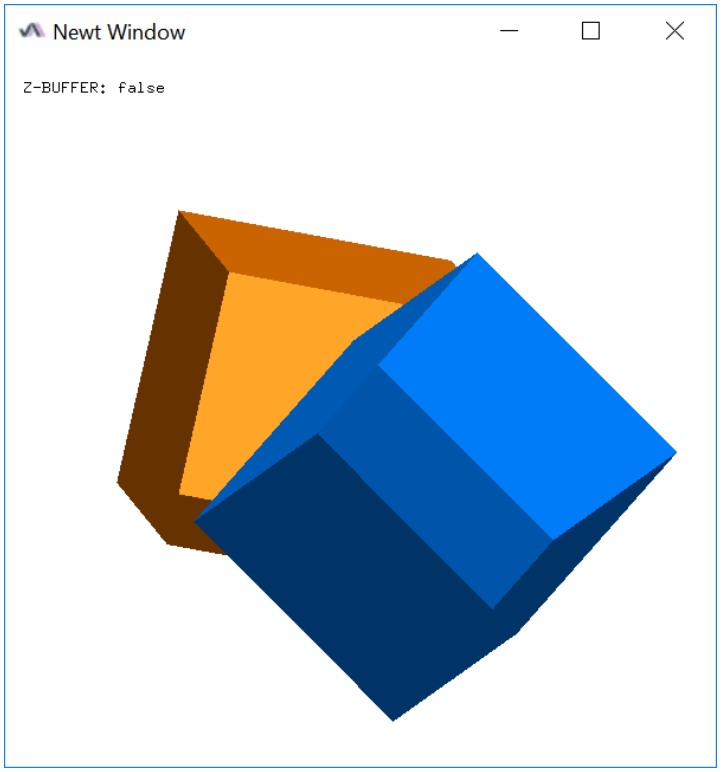


Imagem sem o Z-Buffer ativo

Algoritmo z-buffer

- Lógica do z-buffer
  - Dois buffers do tamanho da janela de exibição são utilizados, um guarda valores de profundidade (`z-buffer`) e o outro guarda valores de cor (`color buffer`)
  - `z-buffer` é iniciado com o maior valor de profundidade possível
  - `color buffer` é iniciado com a cor de fundo da imagem
  - Cada face projetada é percorrida, pixel a pixel, e o valor de profundidade de cada pixel é comparado com o valor já armazenado no `z-buffer`
    - Se este pixel estiver mais próximo do observador, coloca-se seu valor de profundidade no `z-buffer` e seu valor de cor no `color buffer`
    - Assim, no final do processamento o `color buffer` conterá a imagem final

Algoritmo z-buffer

```
1.  Inicialize o plano com a cor de fundo
2.  Inicialize o plano z-buffer com a maior profundidade da cena
3.  para cada polígono
4.          para cada pixel do polígono
5.                  se este for o mais próximo{
6.                          escreva o pixel na tela
7.                          armazena o valor de Z no z-buffer
8.                  }
```

Algoritmo z-buffer

- Configurando o z-buffer no OpenGL
  - Habilita o uso do buffer de profundidade

```
gl.glEnable(GL2.GL_DEPTH_TEST);
```

  - Inicializa os buffers de cor e de profundidade

```
gl.glClear( GL2.GL_COLOR_BUFFER_BIT | GL2.GL_DEPTH_BUFFER_BIT );
```

Modelos de Tonalização (Shading)

- Flat Shading
  - Constante ou facetado – modelo no qual se determina um único valor de intensidade para cada face
  - Colore a face inteira (problema de descontinuidade de intensidade)
  - Vantagem: rapidez na geração da imagem

Pacote tonalizacao

Modelos de Tonalização (Shading)

- Gouraud
  - Obter suavidade na exibição de objetos com superfícies curvas quando representados por faces
  - Elimina parcialmente o problema da descontinuidade
  - Vantagem: melhora do resultado
  - Desvantagem: pontos de brilho especular são um pouco atenuados
- Habilita modelo de tonalização (colorização)

```
gl.glShadeModel(GL2.GL_FLAT);
//ou GL_SMOOTH -> default
```

Iluminação e Tonalização em OpenGL

- Para gerar imagens realistas é necessário:

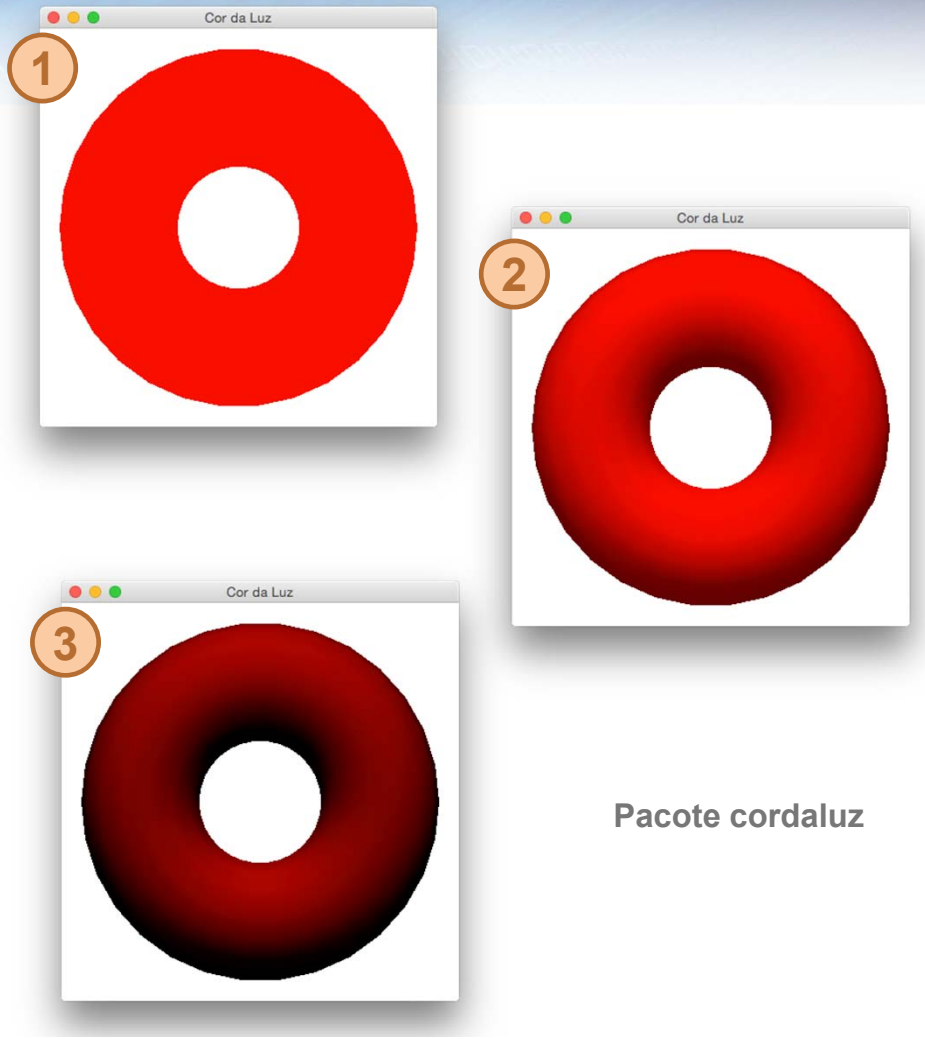
  1. Criar uma ou mais fontes de luz

  2. Escolher os modelos de reflexão e tonalização a serem aplicados

Iluminação e Tonalização em OpenGL

- Componentes de cor de uma luz
  - Quantidade de RGB que a luz emite. Correspondem ao percentual da intensidade total para cada componente

1. Cor da luz (1.0, 1.0, 1.0)
   - A luz será branca e com a maior intensidade possível
2. Cor da luz (0.5, 0.5, 0.5)
   - A cor ainda será branca, mas com metade da intensidade
3. Cor da luz (0.0, 0.0, 1.0)
   - A luz será azul e com intensidade máxima



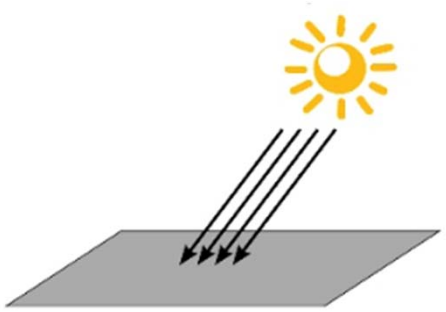
Pacote cordaluz

Iluminação e Tonalização em OpenGL

- Tipos de Luz em OpenGL:
  - ambiente: luz refletida no ambiente, é a luz que vem de todas as direções;
  - difusa: incide em um ponto e se espalha em todas as direções;
  - especular: incide em um ponto e se espalha em uma única direção;

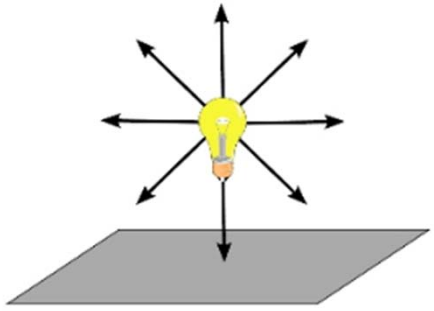
Iluminação e Tonalização em OpenGL

- Posição da Luz:
  - Possui 4 valores (x, y, z, w)
  - x, y e z : coordenadas dos eixos no sistema
  - w : Tratamento da fonte de luz



w = 0.0 – luz direcional que se encontra no infinito, ou a uma distância muito grande.

Exemplo: Sol



w = 1.0 – luz pontual que tem uma posição exata na cena.

Exemplo: Vela, Lâmpada

Iluminação e Tonalização em OpenGL

- Habilitar / Desabilitar o uso de iluminação

```
gl.glEnable(GL2.GL_LIGHTING);
gl.glDisable(GL2.GL_LIGHTING);
```

- É possível ter em uma cena até 8 luzes diferentes
- Habilitar / Desabilitar uma luz

```
gl.glEnable(GL2.GL_LIGHT0);
gl.glDisable(GL2.GL_LIGHT0);
   …..
gl.glEnable(GL2.GL_LIGHT7);
gl.glDisable(GL2.GL_LIGHT7);
```

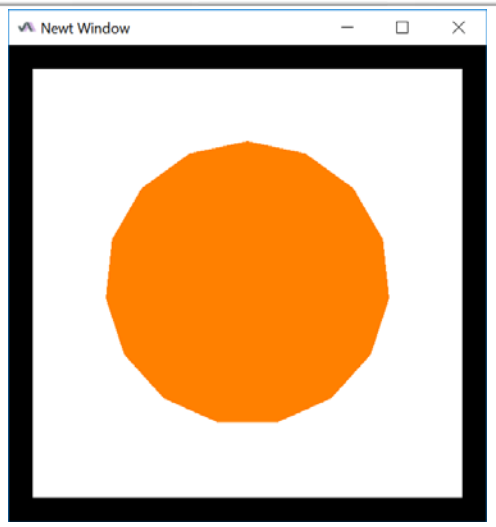
Método ligaLuz() e desligaLuz()

```
public void ligaLuz(GL2 gl) {
    // habilita a definição da cor do material a partir da cor corrente
    gl.glEnable(GL2.GL_COLOR_MATERIAL);

    // habilita o uso da iluminação na cena
    gl.glEnable(GL2.GL_LIGHTING);
    // habilita a luz de número 0
    gl.glEnable(GL2.GL_LIGHT0);
    //Especifica o Modelo de tonalizacao a ser utilizado
    //GL_FLAT -> modelo de tonalizacao flat
    //GL_SMOOTH -> modelo de tonalização GOURAUD (default)
    gl.glShadeModel(tonalizacao);
}
```

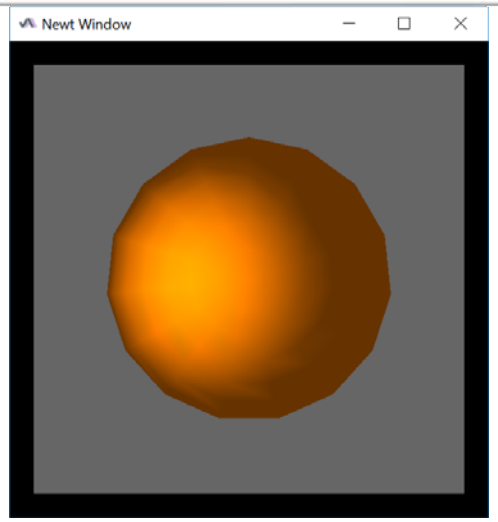
```
public void desligaluz(GL2 gl) {
    //desabilita o ponto de luz
    gl.glDisable(GL2.GL_LIGHT0);
    //desliga a iluminacao
    gl.glDisable(GL2.GL_LIGHTING);
}
```

Luz Ambiente

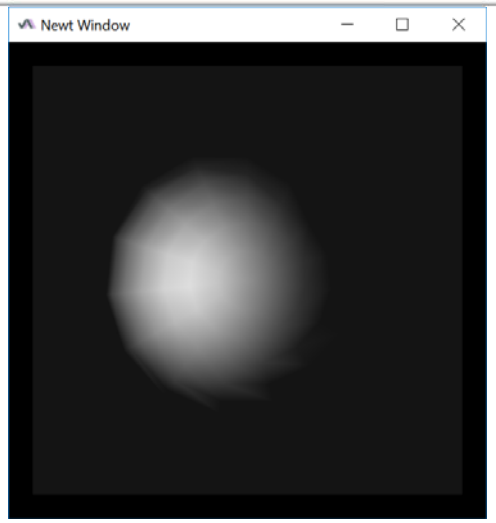
```
public void iluminacaoAmbiente(GL2 gl) {
    float luzAmbiente[] = {0.2f, 0.2f, 0.2f, 1.0f}; //cor
    float posicaoLuz[] = {-50.0f, 0.0f, 100.0f, 1.0f}; //pontual

    // define parametros de luz de número 0 (zero)
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_AMBIENT, luzAmbiente, 0);
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_POSITION, posicaoLuz, 0);
}
```



Luz desligada



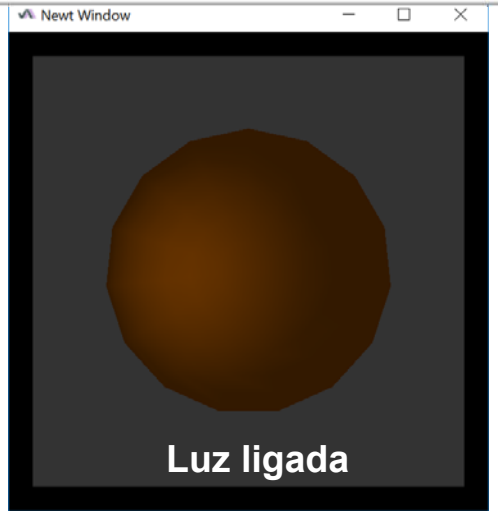
Luz ligada



Cor do Material desabilitada

Luz Difusa

```
public void iluminacaoDifusa(GL2 gl) {
    float luzDifusa[] = {0.2f, 0.2f, 0.2f, 1.0f}; //cor
    float posicaoLuz[] = {-50.0f, 0.0f, 100.0f, 1.0f}; //1.0 pontual

    //define os parâmetros de luz de número 0 (zero)
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_DIFFUSE, luzDifusa, 0);
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_POSITION, posicaoLuz, 0);
}
```



Luz Especular

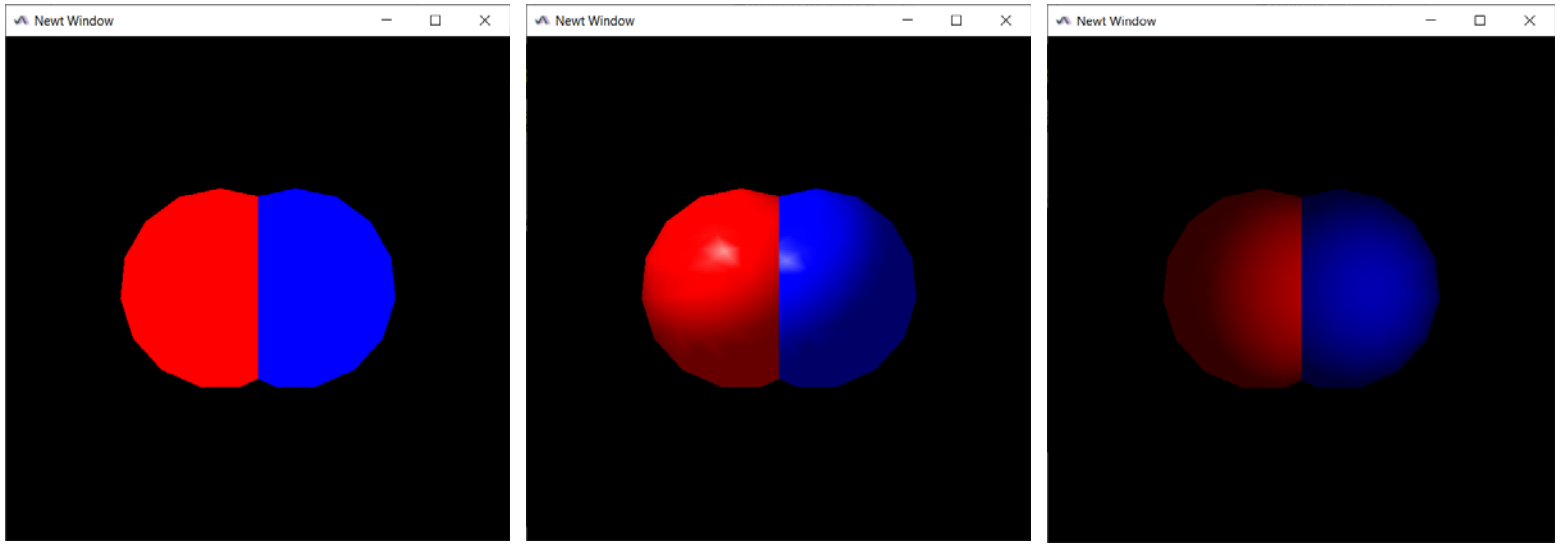
```
public void iluminacaoEspecular(GL2 gl){
    float luzAmbiente[] = {0.2f, 0.2f, 0.2f, 1.0f};
    float luzEspecular[] = {1.0f, 1.0f, 1.0f, 1.0f};
    float posicaoLuz[] = {-50.0f, 0.0f, 100.0f, 1.0f};

    // intensidade reflexao do material
    int especMaterial = 128;
    // define a concentracao do brilho
    gl.glMateriali(GL2.GL_FRONT, GL2.GL_SHININESS, especMaterial);

    // define a reflectancia do material
    gl.glMaterialfv(GL2.GL_FRONT, GL2.GL_SPECULAR, luzEspecular, 0);

    // define os parametros de luz de numero 0 (zero)
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_AMBIENT, luzAmbiente, 0);
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_SPECULAR, luzEspecular, 0);
    gl.glLightfv(GL2.GL_LIGHT0, GL2.GL_POSITION, posicaoLuz, 0);
}
```

Luz Especular

KEEP
CALM
AND
VAMOS
PRATICAR

Exercício

- Crie duas esferas, uma vermelha e outra azul;
- A cena deve permitir:
  - ligar/desligar a iluminação
  - Alternar entre luz especular e difusa
  - Os tratamentos de eventos estão no próximo slide ➔



Exercício

- Tratar os eventos:
  - Tecla r
    - Rotacionar o sorvete em 45° nos eixos x e z
  - Teclar ESC
    - Finalizar a aplicação
  - Tecla w
    - Alternar entre Wire/Solid
  - Tecla L
    - Habilitar / Desabilitar o uso de iluminação
  - Tecla t
    - Alternar entre tonalização GL_FLAT / GL_SMOOTH
  - Tecla 0
    - Habilita iluminação especular
  - Tecla 1
    - Habilita iluminação difusa

Material elaborado por:

Prof. Ms. Simone de Abreu

siabreu@gmail.com

Prof. Dr. Fernando Kakugawa

fernando.kakugawa@animaeducacao.com.br