OpenGL

OpenGL

- O que será abordado:
  - Introdução
  - Objetos geométricos
  - Visualização
  - Cor
  - Iluminação
  - Framebuffer

OpenGL - Introdução (1)

- História
  - Primeiros padrões gráficos
    - PostScript page description language
      - Texto e gráficos 2D
    - PHIGS (*Programmer's Hierarchical Interactive Graphics System*)
      - Baseada no GKS (G*raphics Kernel System*)
      - Descrições de objetos 3D em uma *display list*
    - Iris GL
      - Interface gráfica independente de hardware

OpenGL - Introdução (2)

- OpenGL
  - Sucessora da biblioteca gráfica IRIS GL, ambas desenvolvidas pela Silicon Graphics, Inc. (SGI).
- Órgão mantenedor
  - ARB (*OpenGL Architecture Review Board*)
    - Consórcio independente que administra o uso da OpenGL, formada por diversas empresas da área.

OpenGL - Introdução (3)

- O que é a OpenGL?
  - "GL" significa *Graphics Library*
  - Interface de programação para aplicações gráficas 2D ou 3D
  - Independente do sistema de janelas
    - GLX (*OpenGL Extension for X*)
    - GLUT (*OpenGL Utility Toolkit)*
  - Palavras reservadas começam com `gl` ou `GL_`
  - Várias implementações (SGI, MS, Mesa)

OpenGL - Introdução (4)

- Principais pontos
  - Máquina de estados
    - Vértices são afetados pelo estado atual
      - Matrizes de transformação, cor, iluminação, textura
  - Controle encapsulado
    - Iluminação
    - Textura
  - Interface procedural
    - Programador especifica exatamente como a cena será renderizada
  - Utilizada em várias linguagens (e.g. C, C+)

OpenGL - Introdução (5)

- Sintaxe das rotinas

`glVertex3fv` ← **v** se estiver presente, indica formato vetorial

tipo de dado: **f** float
d double float
s signed short integer
i signed integer

número de componentes: **2**, **3** ou **4**

- Outros tipos de dados nas rotinas da OpenGL

b byte
ub unsigned byte
us unsigned short integer
u unsigned integer

OpenGL - GLUT (1)

- *OpenGL Utility Toolkit*
  - Um conjunto de rotinas que facilita a construção de aplicações que utilizam a OpenGL
    - Gerenciamento de janelas
    - Produção de objetos gráficos 3D
    - Controle de eventos
  - Palavras reservadas começam com `glut`
  - Executável em várias plataformas

OpenGL - GLUT (2)

- Principais rotinas para gerenciamento de janelas
  - Inicializa e seleciona atributos da janela (e.g. dimensões, buffers, modo de cor)
    - `glutInit()`
    - `glutInitDisplayMode()`
    - `glutInitWindowSize()`
    - `glutInitWindowPosition()`
  - Cria a janela e o contexto de renderização
    - `glutCreateWindow()`

OpenGL - GLUT (3)

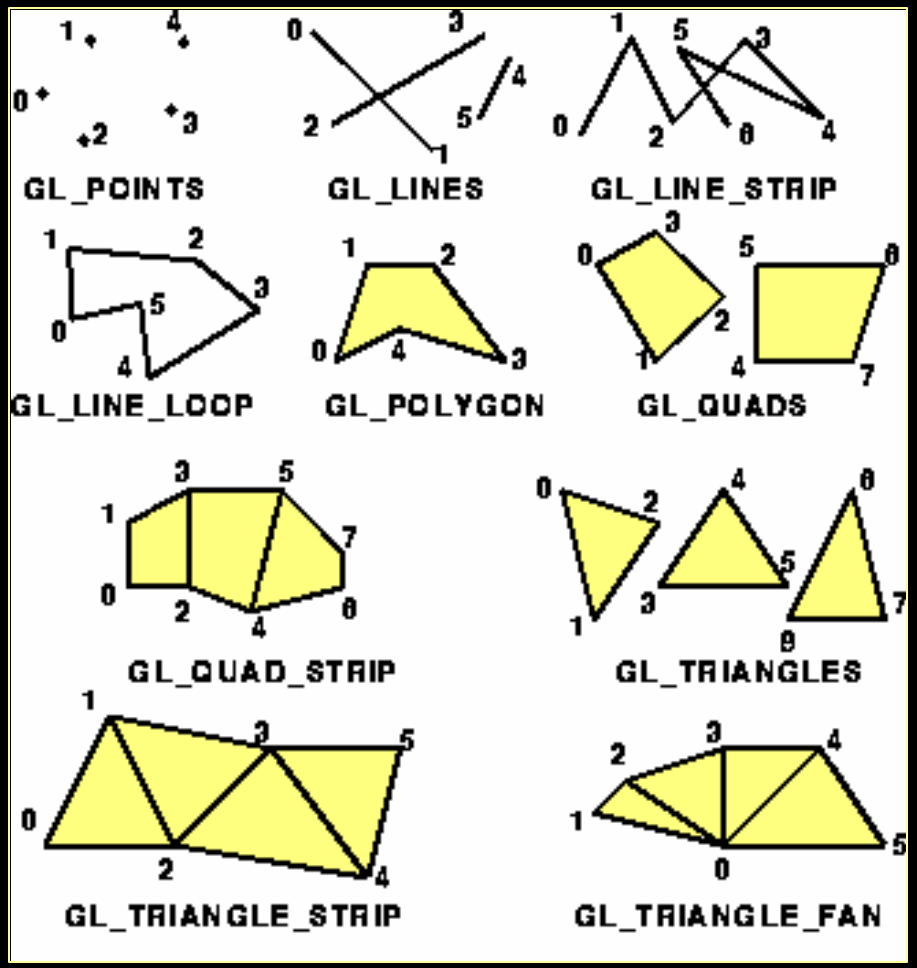
- Principais rotinas para gerenciamento de janelas (continuação)
  - Define funções a serem executadas quando a janela é alterada
    - `glutReshapeFunc()`
    - `glutDisplayFunc()`
  - Processamento de eventos (loop infinito)
    - `glutMainLoop()`

OpenGL - Objetos geométricos (1)

- A OpenGL não possui rotinas de alto nível de abstração.
- As primitivas geométricas são definidas a partir de seus vértices, através das rotinas `glVertex*()`.
- Os vértices são agrupados entre as rotinas `glBegin()` e `glEnd()`. O argumento da `glBegin()` indica a ordem como os vértices são associados.

OpenGL - Objetos geométricos (2)

Primitivas geométricas da OpenGL



OpenGL - *Display list* (1)

- Encapsula um conjunto de rotinas da OpenGL para serem executadas posteriormente.
- Vantagem
  - Funções e variáveis são avaliadas e substituídas por seus respectivos valores
- Desvantagem
  - Não é possível modificar uma *display list*

OpenGL - *Display list* (2)

- É definida agrupando as instruções entre as funções `glNewList()` e `glEndList()`.
- Uma *display list* é executa através da rotina `glCallList()`. O parâmetro identifica a *display list* a ser executada.

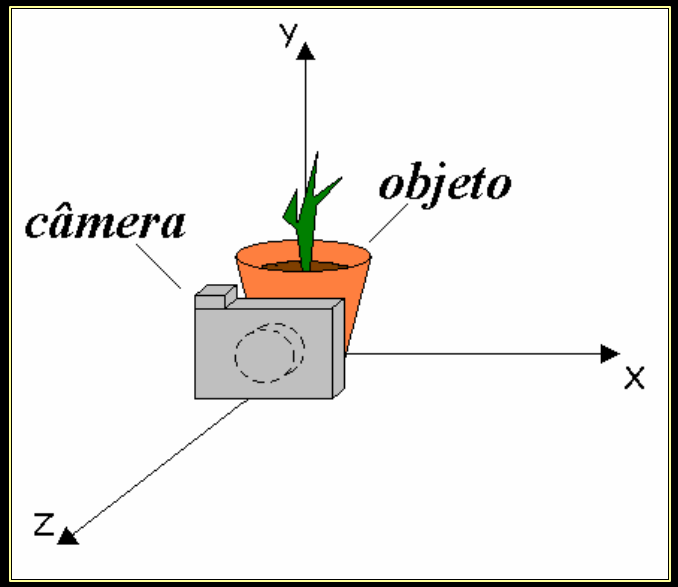
OpenGL - Visualização (1)

- São operações necessárias para converter objetos definidos em um espaço 3D para um espaço 2D (tela do computador).
  - Aspectos a serem considerados:
    - Transformações
    - *Viewport*

OpenGL - Visualização (2)

- Modelagem e visualização
  - Câmera e objetos são originalmente posicionados na origem

rotinas
- `glTranslate*()`
- `glRotate*()`
- `glScale*()`

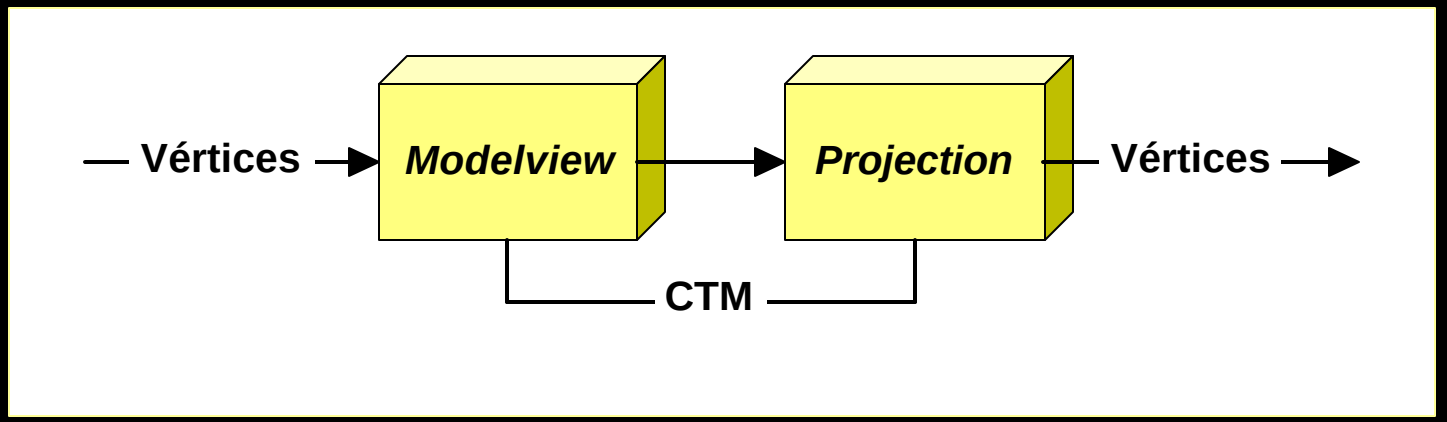


OpenGL - Visualização (3)

- Operações com as matrizes de transformação são realizadas através de duas pilhas:
  - *Modelview* ⇨ modelagem e visualização
  - *Projection* ⇨ projeção
- Por que utilizar a metodologia de pilha?
  - Organizar a seqüência de operações sobre as matrizes de transformação
  - Facilitar a construção de modelos hierárquicos

OpenGL - Visualização (4)

- Matriz de transformação atual (CTM - *Current Transformation Matrix)*



- *Viewport*
  - Através da rotina `glViewport()`, define uma correspondência entre as vértices transformados e os pixels da tela.

OpenGL - Visualização (5)

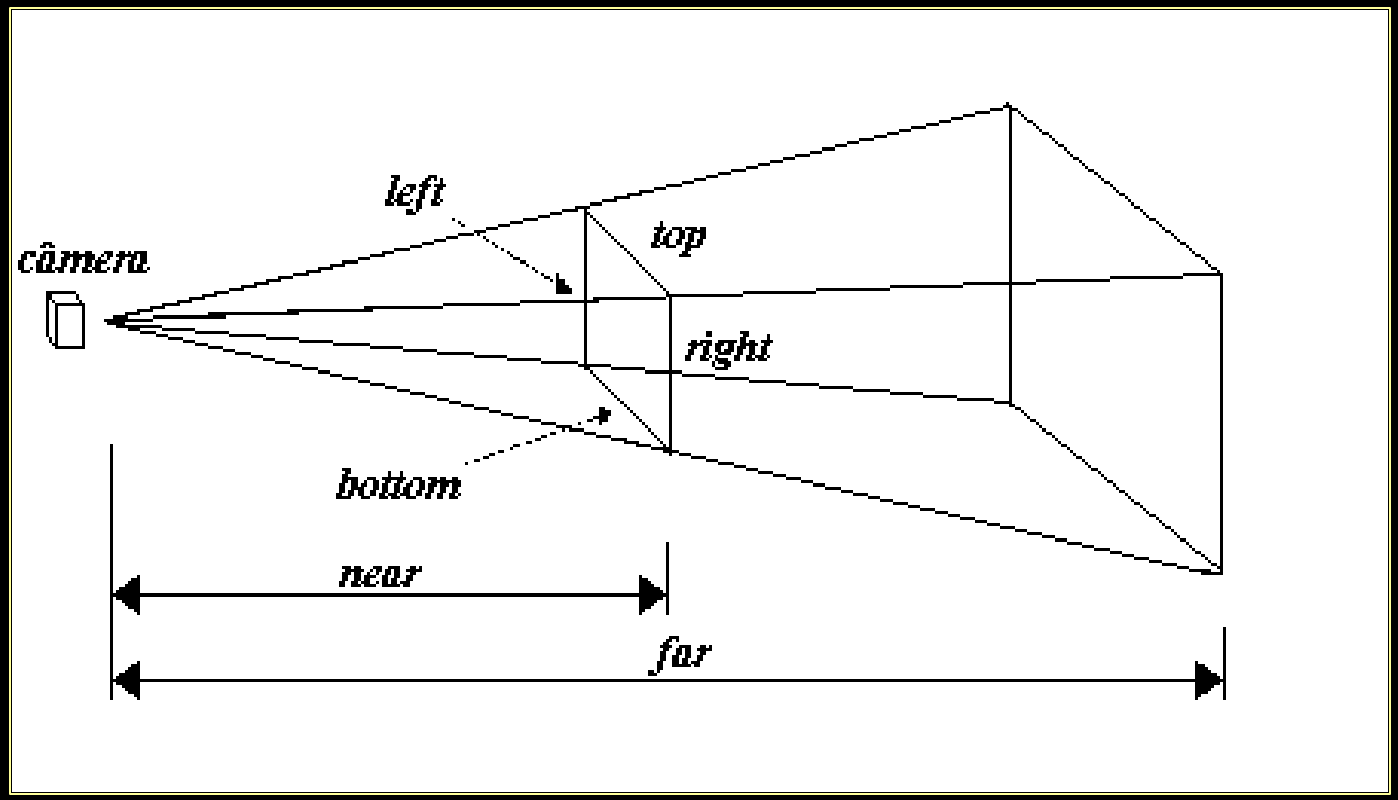
- A rotina `glMatrixMode()` especifica qual é a pilha a ser modificada, através dos argumentos `GL_MODELVIEW` ou `GL_PROJECTION`
- O controle sobre a pilha é realizado pelas rotinas `glPushMatrix()` e `glPopMatrix()`

OpenGL - Visualização (6)

- Projeção
  - A OpenGL provê dois tipos de projeção para definir o volume de visualização:
    - Perspectiva
    - Ortogonal

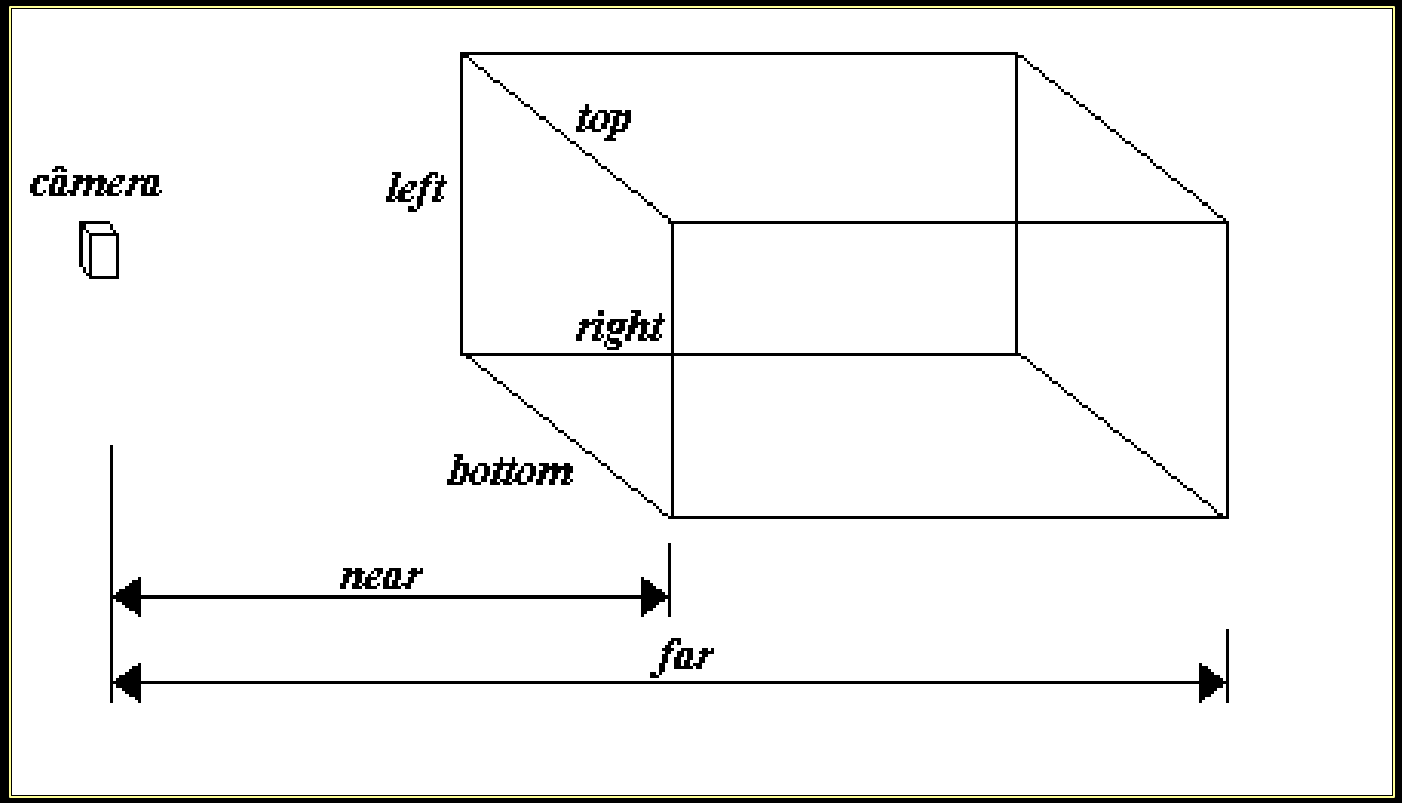
OpenGL - Visualização (7)

- Projeção perspectiva: através da rotina `glFrustum()`, define um volume de visualização onde a projeção do objeto é reduzida a medida que ele é afastado da câmera.



OpenGL - Visualização (8)

- Projeção ortogonal: através da rotina `glOrtho()`, define um volume de visualização onde a projeção do objeto não é afetada pela sua distância em relação à câmera.



OpenGL - Cor (1)

- Possui dois modos de cor:
  - RGBA *(Red, Green, Blue, Alpha)*
  - Indexado
- A seleção do modo de cor é feita pela interface responsável pelo gerenciamento das janelas. Na GLUT, a seleção do modo de cor é feita pela rotina `glutInitDisplayMode()`.

OpenGL - Cor (2)

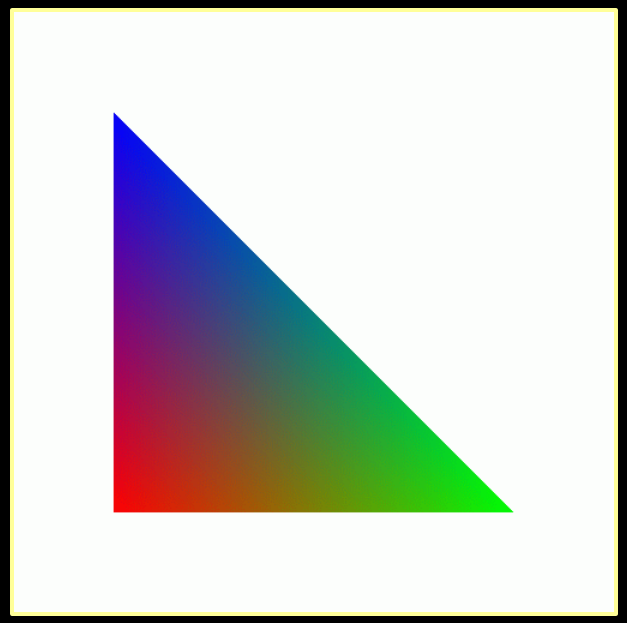
- RGBA
  - As componentes variam de 0 a 1
  - A componente *alpha* é utilizada, e. g., em operações de transparência
  - As rotinas glColor*() são utilizadas para definir cada componente

OpenGL - Cor – Exemplo (3)

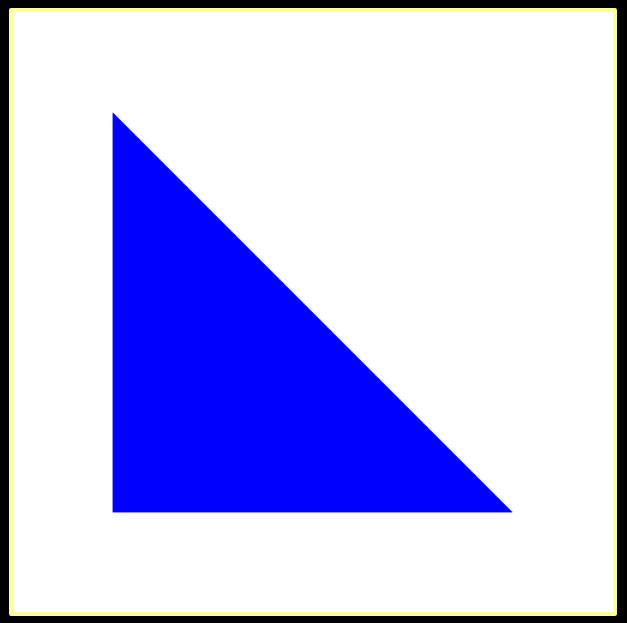
- Exemplo (RGBA)

```
glShadeModel(GL_SMOOTH);
glBegin(GL_TRIANGLES);
    glColor3f(1.0,0.0,0.0);
    glVertex2f(5.0,5.0);
    glColor3f(0.0,1.0,0.0);
    glVertex2f(25.0,5.0);
    glColor3f(0.0,0.0,1.0);
    glVertex2f(5.0,25.0);
glEnd();
```

OpenGL - Cor – Exemplo (4)



GL_SMOOTH



GL_FLAT

O modo de preenchimento é definido pela rotina `glShadeModel()`, através do argumento `GL_SMOOTH` (método *Gouraud Shading)* ou `GL_FLAT` (a cor de um vértice é utilizado para o preenchimento)

OpenGL - Iluminação (1)

- A OpenGL utiliza o modelo de iluminação de *Phong*

Necessário especificar

- Propriedades dos materiais
  - Vetores normais
  - Componentes (emitida, ambiente, difusa e especular)
- Fontes de luz
  - Tipo (direcional, pontual, *spot* e ambiente)
  - Componentes
  - Posição

OpenGL - Iluminação (2)

- Componentes de iluminação
  - Emitida: proveniente de um objeto
  - Ambiente: sem fonte de luz específica, proveniente das várias reflexões nas superfícies da cena
  - Difusa: proveniente de uma direção específica, reflete em todas as direções

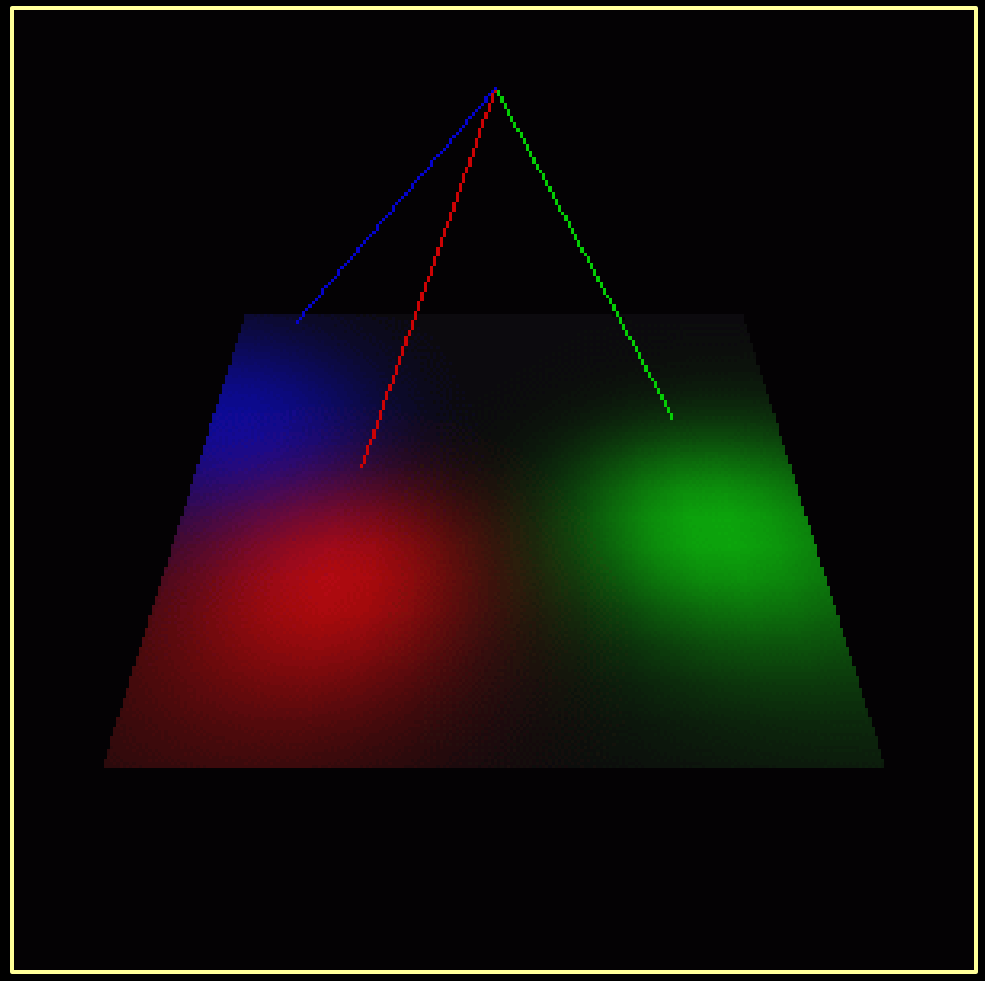
OpenGL - Iluminação (3)

- Componentes de iluminação (continuação)
  - Especular: proveniente de uma direção específica, reflete em uma direção específica
- Obs: componentes ambiente e difusa definem a cor do objeto e a componente especular define o brilho

OpenGL - Iluminação (4)

- Fontes de luz
  - Fonte direcional: irradia energia luminosa em uma determinada direção
  - Fonte pontual: irradia energia luminosa em todas direções
  - Fonte *spot*: tem uma direção principal na qual ocorre a máxima concentração de energia
- As rotinas `glLight*()` são utilizadas para especi-ficar o tipo, a posição e as componentes da fonte de luz.

OpenGL – Iluminação - Exemplo (5)



Fontes do tipo *spot*

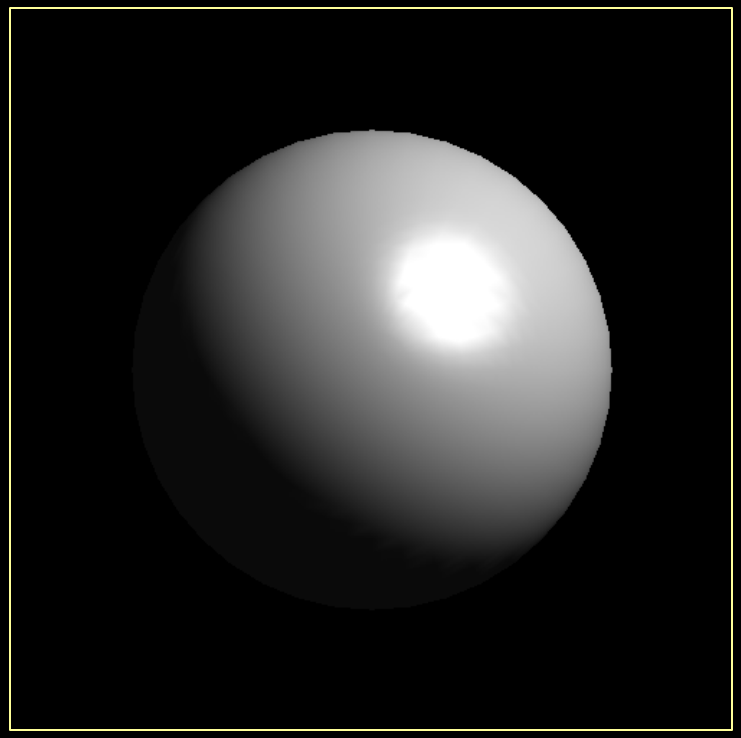
OpenGL - Iluminação (6)

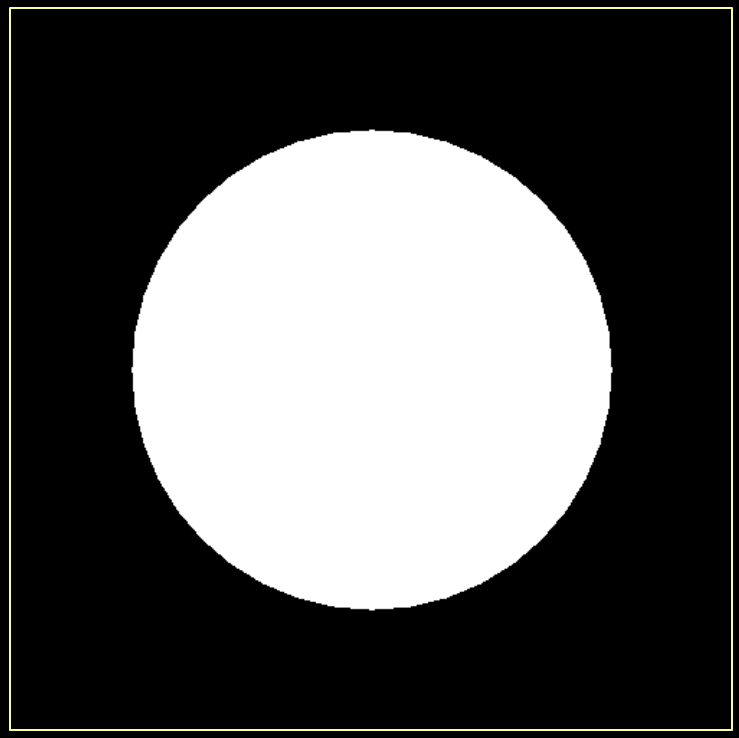
- Materiais
  - As rotinas `glMaterial*()` são utilizadas para definir as propriedades dos materiais especificando os valores RGB para cada componente de iluminação
  - As rotinas `glNormal*()` determinam a orientação relativa do objeto em relação às fontes de luz

OpenGL - Iluminação (7)

- Algumas considerações
  - As rotinas `glLightModel*()` descrevem os parâmetros de um modelo de iluminação como, e.g., intensidade da luz ambiente e *viewpoint*
  - As rotinas `glEnable(GL_LIGHTING)` e `glEnable(GL_LIGHT`*`i`*`)` respectivamente habilitam o modelo de iluminação e uma determinada fonte de luz

OpenGL - Iluminação (8)





OpenGL - Framebuffer (1)

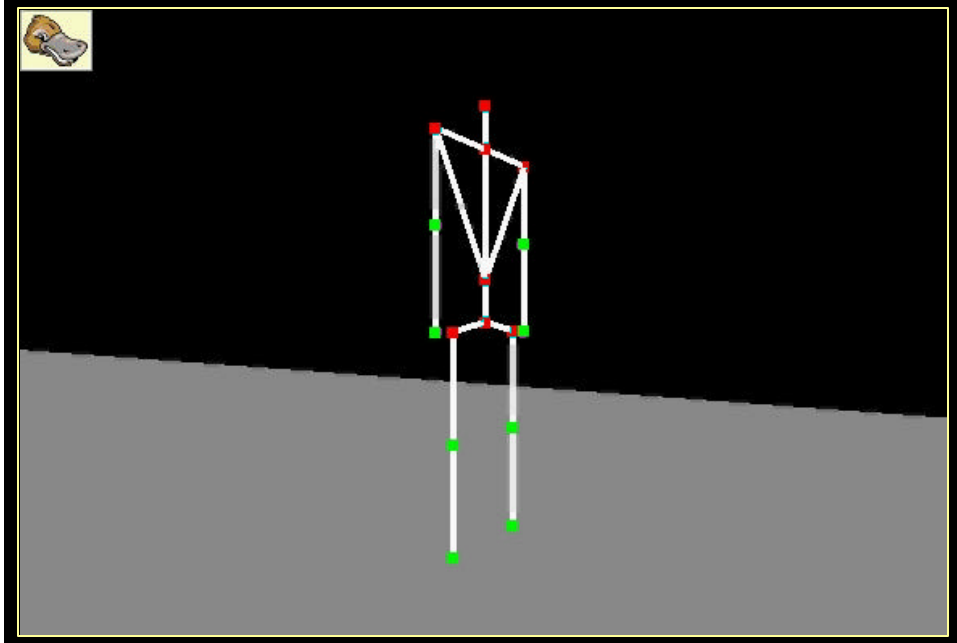
- Um conjunto de buffers que possui funções específicas
  - Buffer de cor
  - Buffer de profundidade
  - Buffer *stencil* (seleção)
  - Buffer de acumulação

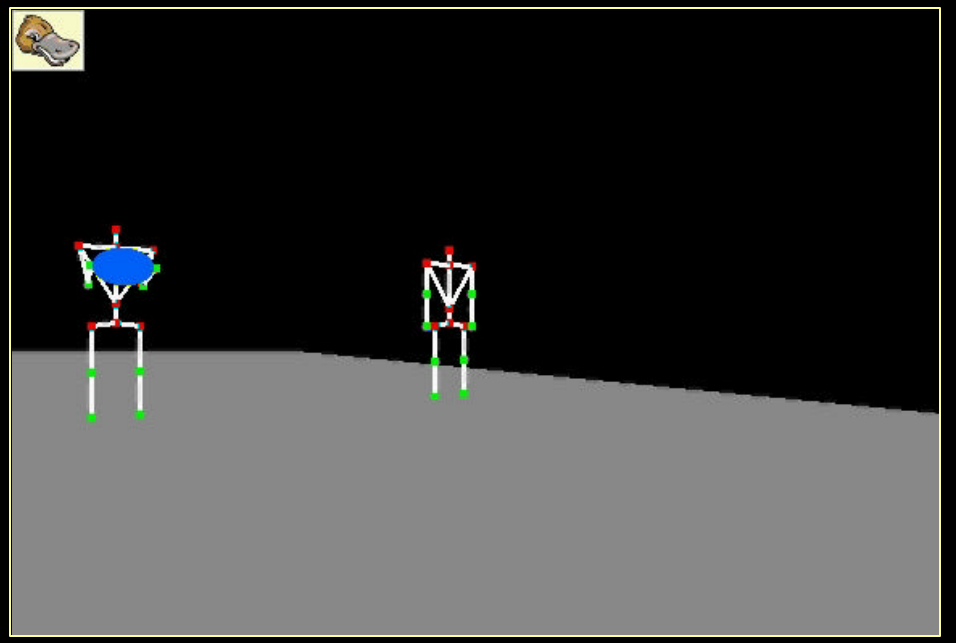
OpenGL - Framebuffer (2)

- Buffer de cor
  - É o buffer utilizado para renderizar a imagem. Em alguns casos, como em animações, pode-se utilizar um buffer de cor auxiliar (*double buffer*).

OpenGL - Framebuffer (3)

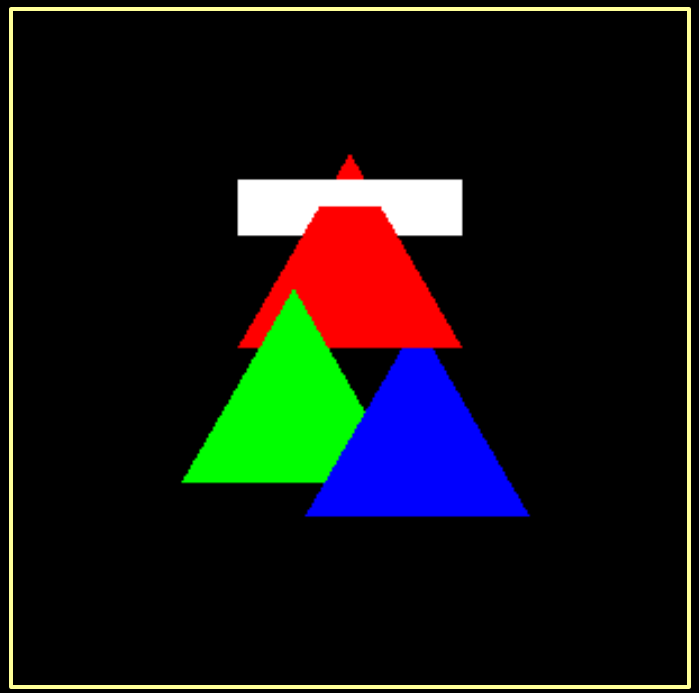
- Buffer de cor (exemplo *double buffer)*





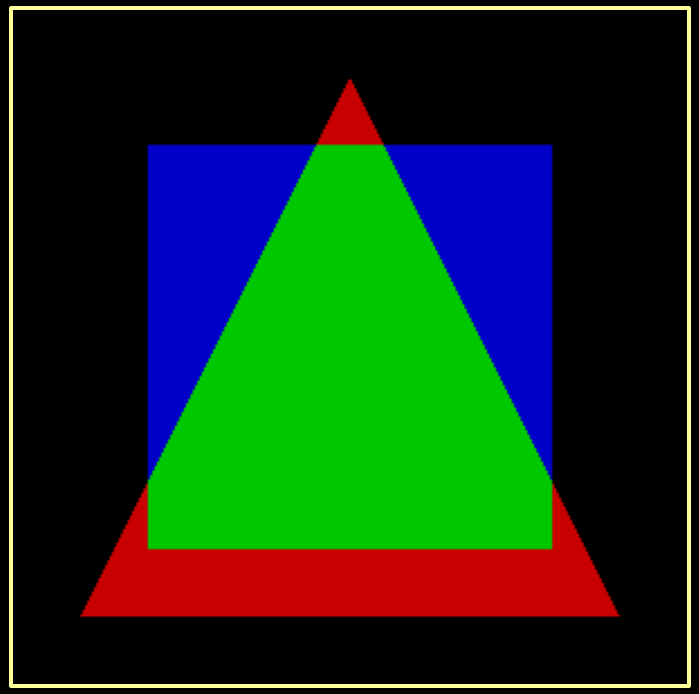
OpenGL - Framebuffer (4)

- Buffer de profundidade
  - É o buffer utilizado para armazenar a coordenada *z* de cada pixel (algoritmo *z buffer*).



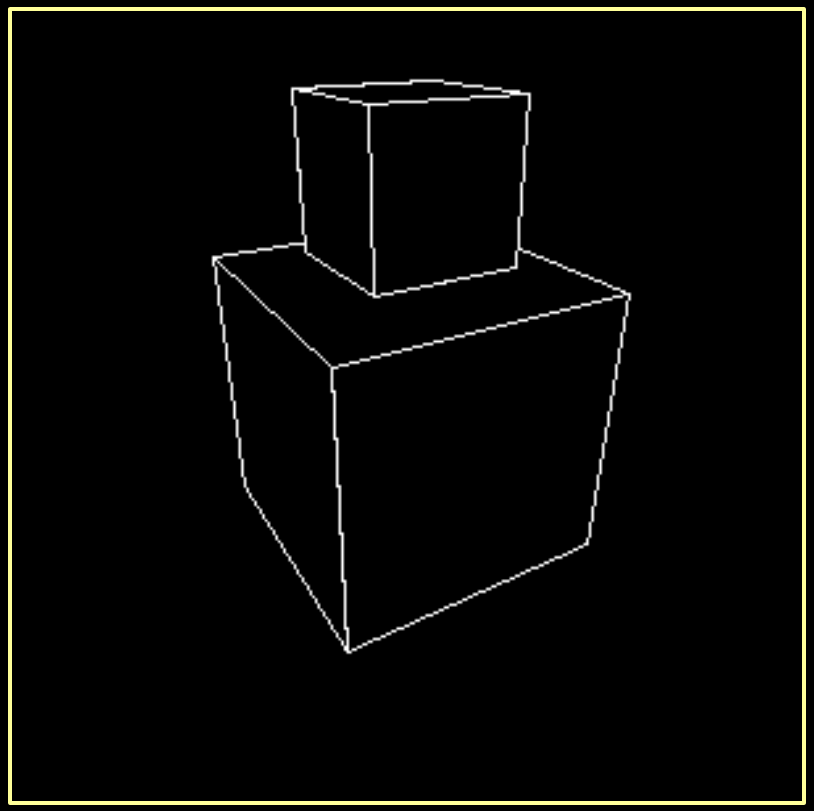
OpenGL - Framebuffer (5)

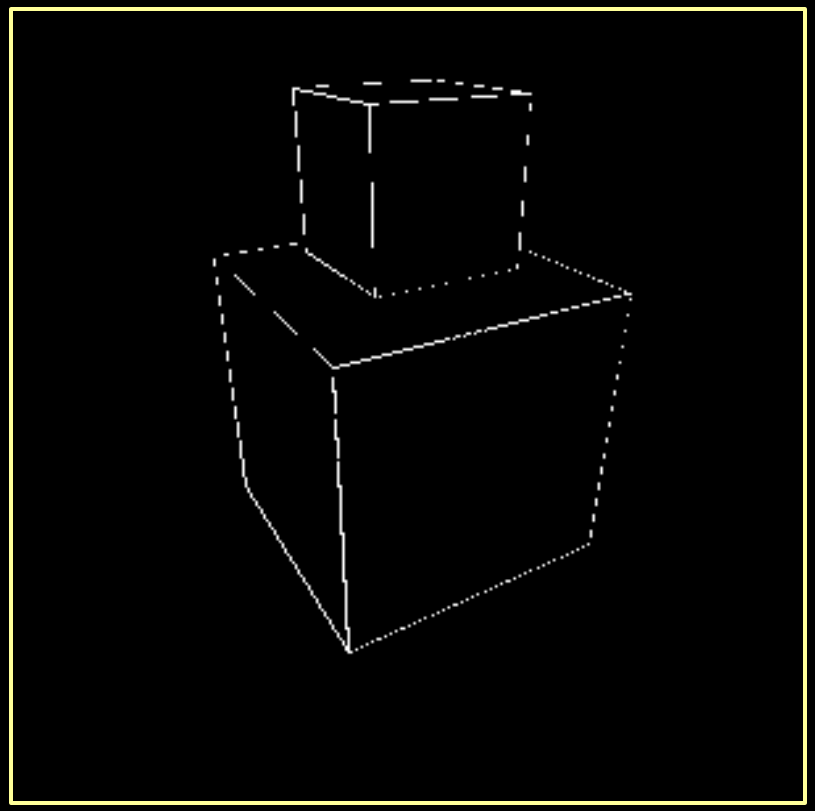
- Buffer stencil (seleção)
  - É utilizado para selecionar determinadas áreas, conforme alguns testes.



OpenGL - Framebuffer (6)

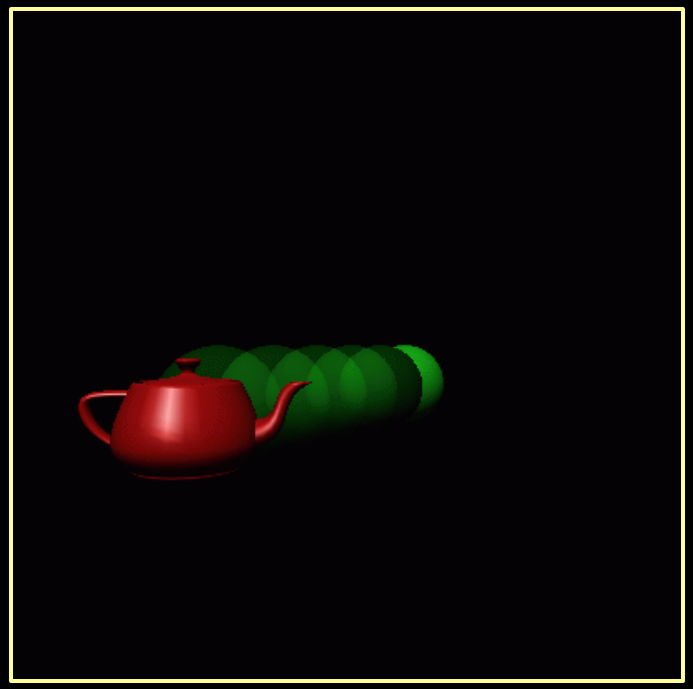
- Buffer stencil (seleção)

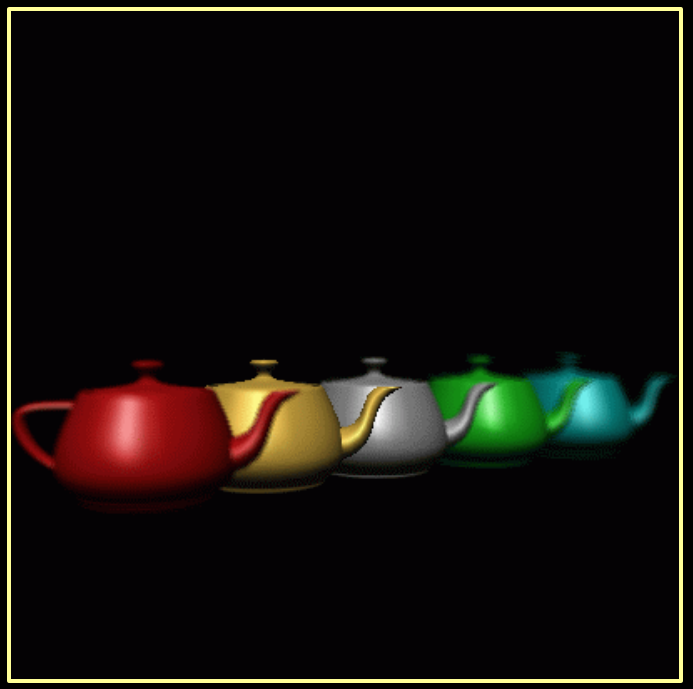




OpenGL - Framebuffer (7)

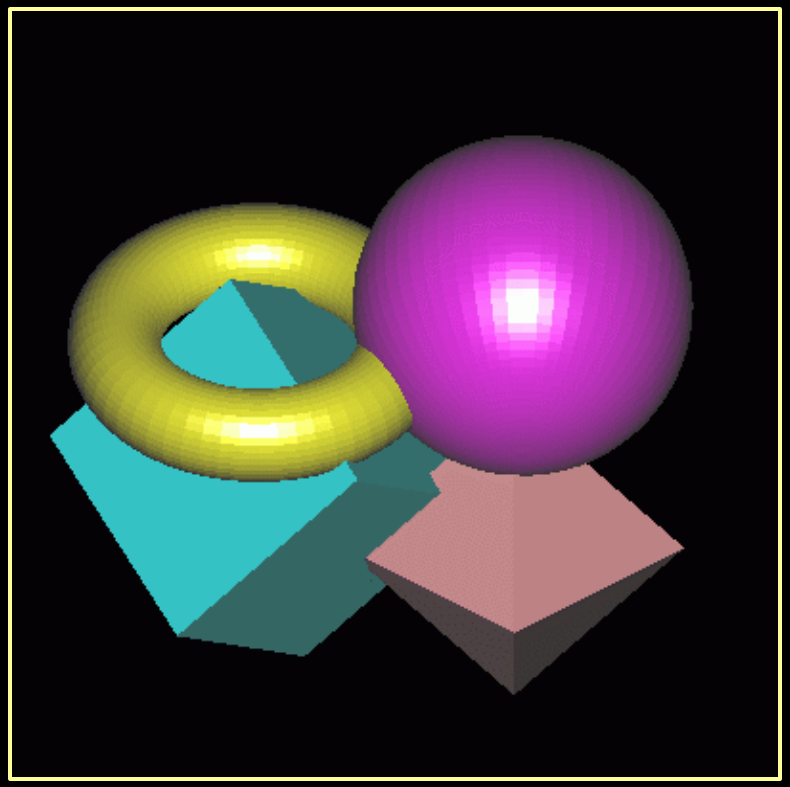
- Buffer de acumulação
  - É utilizado para trabalhar com diversas técnicas, como *antialiasing*, *montion blur* e campo de profundidade.

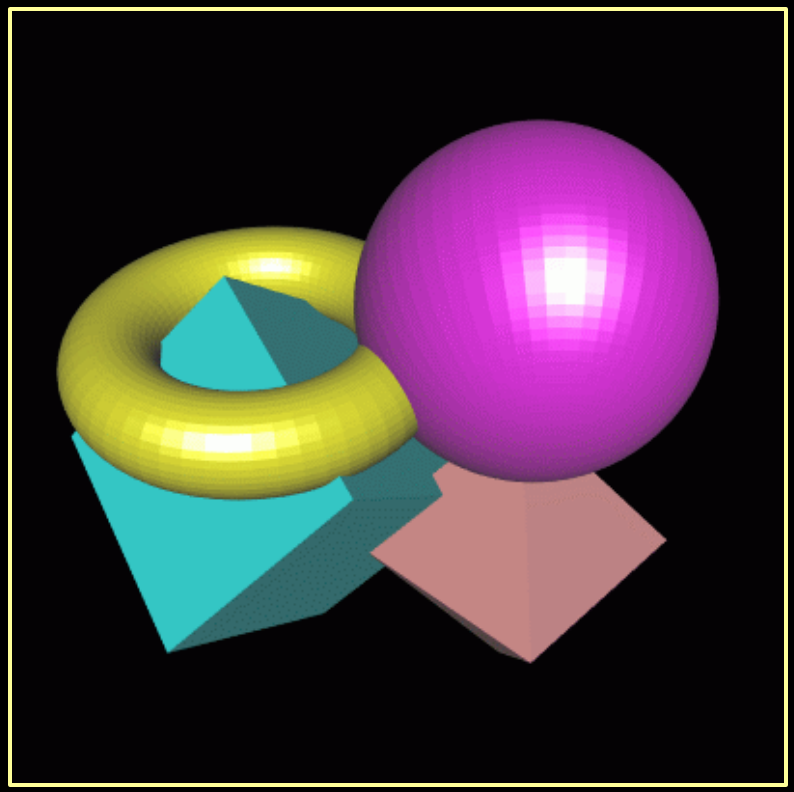




OpenGL - Framebuffer (8)

- Buffer de acumulação





OpenGL

FIM!

Profa. Wu, Shing Ting
ting@dca.fee.unicamp.br
Alessandro Bicho
bicho@dca.fee.unicamp.br
Harlen Batagelo
harlen@dca.fee.unicamp.br