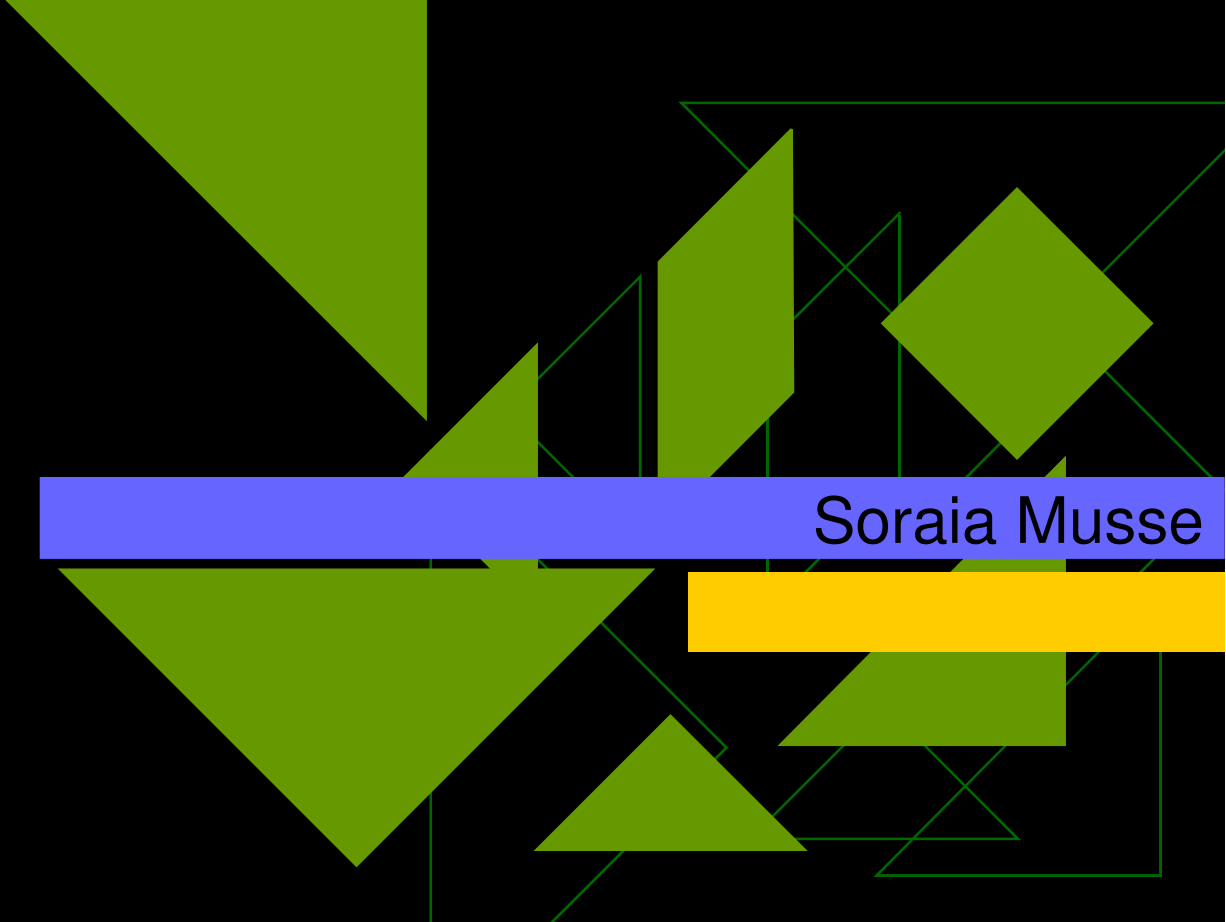


OpenGL



Soraia Musse

Roteiro

- ◆ **Introdução**
- ◆ Utilização
- ◆ Exemplo de Programa
- ◆ Nomes das Funções/Métodos
- ◆ Bibliotecas
- ◆ Máquina de Estados
- ◆ Linhas, Pontos e Polígonos
- ◆ Transformações Geométricas

Introdução

- ◆ OpenGL
 - Biblioteca de rotinas gráficas e de modelagem, 2D e 3D (interface para hardware gráfico)
- ◆ API (*Application Programming Interface*) padrão para desenvolvimento de aplicações gráficas 3D em tempo real
- ◆ Portável
- ◆ Rápida
- ◆ Grande qualidade visual, dá suporte para:
 - Mapeamento de textura
 - Iluminação
 - Transparência
 - Animação
 - Outros efeitos especiais

Introdução

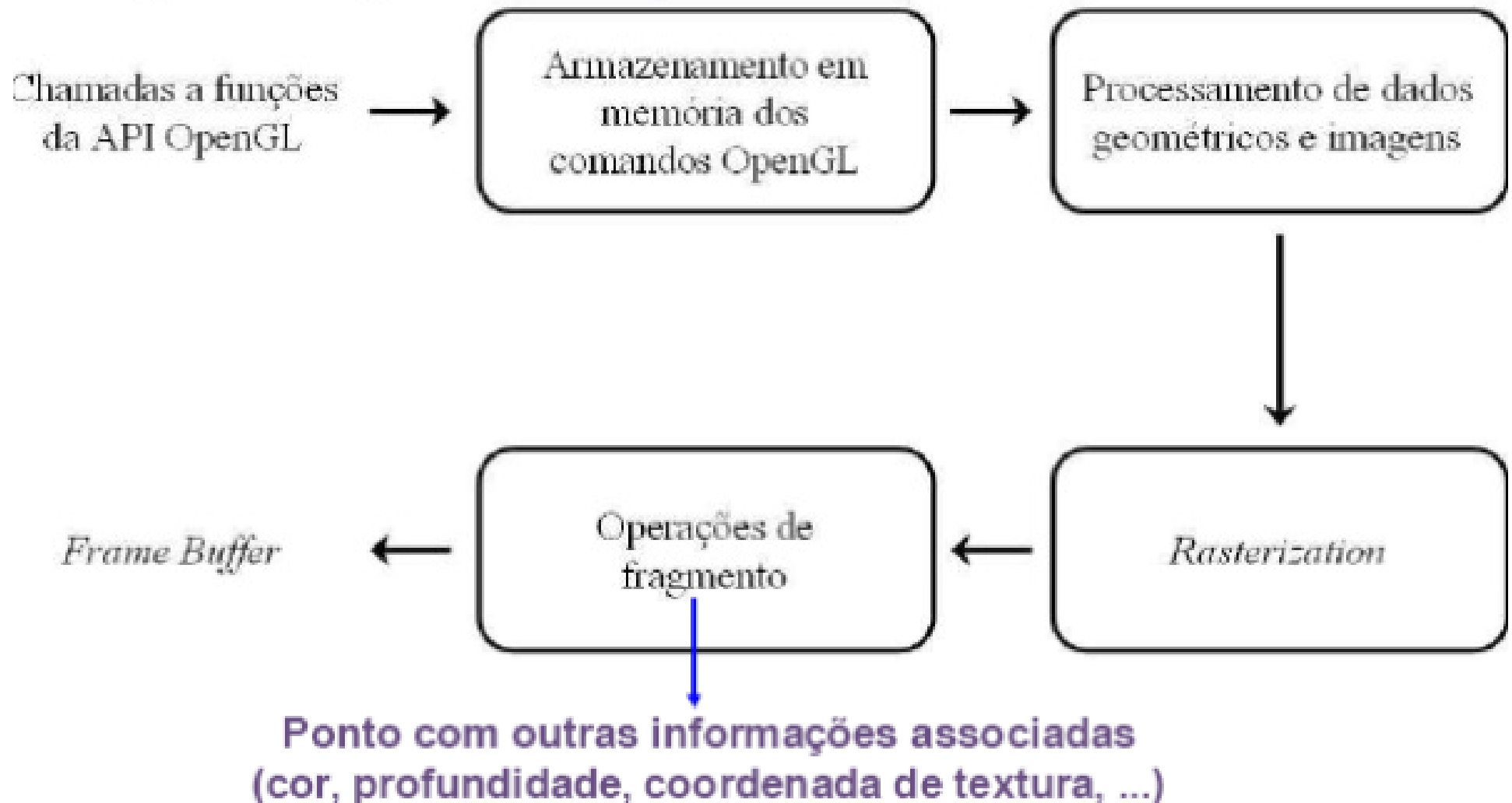
- ◆ OpenGL
- ◆ Aproximadamente 250 comandos e funções
 - 200 comandos do core OpenGL
 - 50 comandos da GLU (*OpenGL Utility Library*)
 - **GLUT** (OpenGL Utility Toolkit) é uma biblioteca de funcionalidades para [OpenGL](#)
- ◆ Programas são baseados em OpenGL ou são aplicações OpenGL (escritos em alguma linguagem de programação que faz chamadas a uma ou mais bibliotecas OpenGL)

Introdução

- ◆ OpenGL
- ◆ Ao invés de descrever a cena e como ela deve parecer, é preciso apenas determinar os passos necessários para alcançar uma certa aparência ou efeito
- ◆ Por ser portátil, OpenGL não possui funções para gerenciamento de janelas, interação com o usuário ou arquivos de entrada/saída
- ◆ A palavra *pipeline* é usada para descrever um processo que pode ter dois ou mais passos distintos

Pipeline OpenGL

■ Pipeline OpenGL simplificado



Utilização

- ◆ OpenGL segue a convenção de chamada de C, e foi escrita em C
- ◆ Existem muitas implementações desta biblioteca
- ◆ Para *Windows* e *Linux*
- ◆ Para C/C++, Java, C#, Python, Delphi, ...
- ◆ Para cada implementação são fornecidas as bibliotecas necessárias
- ◆ Por exemplo, para C/C++ no ambiente Windows
 - Bibliotecas *opengl32.lib* (OpenGL) e *glu32.lib* (GLU), arquivos *.h* e arquivos *dll*
- ◆ Por exemplo, a JOGL para Java no ambiente Windows
 - Bibliotecas *gluegen-rt.jar* e *jogl.jar* e arquivos *dll*

Exemplo de um programa

- ◆ Estrutura básica de uma aplicação interativa
 - Configura e abre uma janela
 - Cria um *GLCanvas*, adiciona na janela, e especifica o objeto "ouvinte" para os eventos
 - Cria uma classe para fazer o *rendering* que implementa a interface `GLEventListener`
 - Entra no laço de processamento de eventos

Exemplo:

...

```
public class JanelaExemploJava2d {  
    private Renderer2D renderer;  
    public JanelaExemploJava2d(){  
        // Cria janela  
        JFrame janela =  
            new JFrame("Desenho de um  
triângulo em 2D");  
        janela.setBounds(50,100,500,500);  
    }  
}
```

...

Exemplo:

...

// Cria um objeto GLCapabilities para especificar o número

// de bits por pixel para RGBA

GLCapabilities c = new GLCapabilities();

c.setRedBits(8);

c.setBlueBits(8);

c.setGreenBits(8);

c.setAlphaBits(8);

// Cria o objeto que irá gerenciar os eventos

renderer = new Renderer2D();

...

Exemplo:

...

// Cria um GLCanvas, adiciona na janela, e especifica o
// objeto "ouvinte" para os eventos GL de teclado

```
GLCanvas canvas = new GLCanvas(c);  
janela.add(canvas, BorderLayout.CENTER);  
canvas.addGLEventListener(renderer);  
canvas.addKeyListener(renderer);  
janela.setVisible(true);  
} // Fim do construtor
```

```
public static void main(String args[]) {  
    JanelaExemploJava2d ej = new JanelaExemploJava2d();  
}  
}  
...
```

Exemplo de Programa

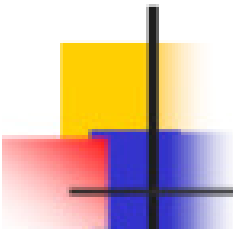
```
...
// Método definido na interface GLEventListener e chamado pelo
// objeto no qual será feito o desenho para começar a fazer o
// desenho OpenGL pelo cliente.
public void display(GLAutoDrawable drawable)
{
    gl.glClear(GL.GL_COLOR_BUFFER_BIT);
    gl.glLoadIdentity();
    gl.glColor3f(0.0f, 0.0f, 1.0f);

    // Desenha um triângulo no plano xy
    gl.glBegin(GL.GL_TRIANGLES);
        gl.glVertex3d(-0.5, -0.5, 0.0);
        gl.glVertex3d( 0.0, 0.5, 0.0);
        gl.glVertex3d( 0.5, -0.5, 0.0);
    gl.glEnd();
}
...
```

Limpa a janela de visualização com a cor de fundo especificada

Especifica que a cor corrente é azul (R, G, B)

Desenha um triângulo preenchido com a cor corrente e definido pelos vértices especificados a entre *glBegin* e *glEnd*



Exemplo de Programa

...

```
// Método definido na interface KeyListener que está sendo  
// implementado para que seja feita a saída do sistema quando  
// for pressionada a tecla ESC.
```

```
public void keyPressed(KeyEvent e) {  
    switch (e.getKeyCode())  
    {  
        case KeyEvent.VK_ESCAPE: System.exit(0);  
        break;  
    }  
    glDrawable.display();  
}
```

Abandona a execução do programa
se o usuário pressionar ESC.

```
}
```

Nomes das funções/métodos

- Todos os nomes das funções/métodos OpenGL seguem uma convenção que indica
 - De qual biblioteca a função faz parte
 - Quantos e que tipos de argumentos a função tem

■ Exemplo: glColor3f

Prefixo gl
representa a
biblioteca gl

raiz

Sufixo 3f significa que a
função possui três
valores de ponto
flutuante como parâmetro

Nomes das funções/métodos

- ◆ Formato do nome das funções:
<PrefixoBiblioteca><ComandoRaiz><ContArgsOpcional><Tipo
ArgsOpcional>
- ◆ Exemplos:
- ◆ *glColor3i*: recebe três valores inteiros
- ◆ *glColor3d*: recebe três valores double
- ◆ *glColor4i*: recebe quatro valores (RGB e a-transparência)
- ◆ *glutSolidCube(float size)* : função da biblioteca *glut* que recebe um argumento como entrada (tamanho do cubo)
- ◆ *gluLookAt*: função da biblioteca *glu* que recebe nove argumentos como entrada

OpenGL – Máquina de Estados

OpenGL é uma máquina de estados

É possível colocá-la em vários estados, ou modos, que não são alterados, a menos que uma função seja chamada para isto

- Por exemplo, a cor corrente é uma variável de estado que pode ser “*setada*” para branco, de tal maneira que todos os objetos, então, são desenhados com a cor branca, até o momento em que é atribuído outro valor para a cor corrente

OpenGL – Máquina de Estados

OpenGL mantém uma série de variáveis de estado

- Cor
- Estilo (ou padrão) de uma linha
- Posições e características das luzes
- Propriedades do material dos objetos

Muitas variáveis de estado referem-se a modos que podem ser habilitados ou desabilitados com os comandos:

- *glEnable()*
- *glDisable()*

OpenGL – Máquina de Estados

Também é possível salvar e restaurar um conjunto de variáveis de estado em uma pilha usando as seguintes funções

- ◆ void glPushAttrib(GLbitfield mask)
- ◆ void glPopAttrib (void)

Exemplo:

- ◆ glPushAttrib(GL_LINE_BIT); // *Empilha atributo (espessura da linha - default é 1)*
- ◆ glPushAttrib(GL_CURRENT_BIT); // *Empilha atributo (cor)*
- ◆ glColor3f(0.0f, 0.0f, 1.0f); // *Especifica a cor azul*
- ◆ glLineWidth(5); // *Especifica a espessura da linha*
- ◆ // Desenha
- ◆ glPopAttrib(); // *Desempilha atributo (cor)*
- ◆ glPopAttrib(); // *Desempilha atributo (espessura da linha)*

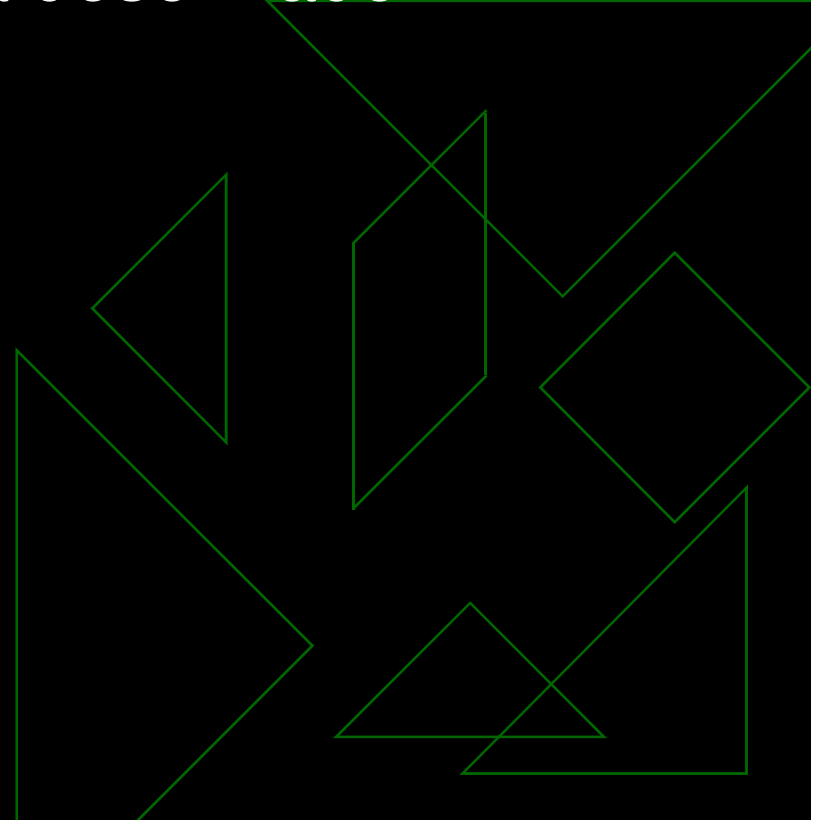
Primitivas geométricas:

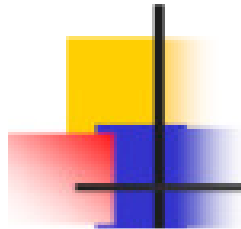
- ◆ Objetos e cenas criados com OpenGL consistem em simples e pequenos *shapes* (ou primitivas, tais como pontos, linhas e polígonos) combinados de várias maneiras
- ◆ OpenGL fornece ferramentas para desenhar pontos, linhas, "polilinhas" e polígonos, que são formados por um ou mais vértices
- ◆ Lista de vértices:
 - `glBegin()`
 - `glVertex2i()` // ou, por exemplo, `glVertex3f()`
 - `glVertex2i()` // ou, por exemplo, `glVertex3f()`
 -
 - `glEnd()`

Primitivas geométricas:

◆ O argumento passado para *glBegin()* determina qual objeto será desenhado:

- ◆ GL_POINTS
- ◆ GL_LINES
- ◆ GL_LINE_LOOP
- ◆ GL_POLYGON
- ◆ GL_TRIANGLES
- ◆ GL_QUAD

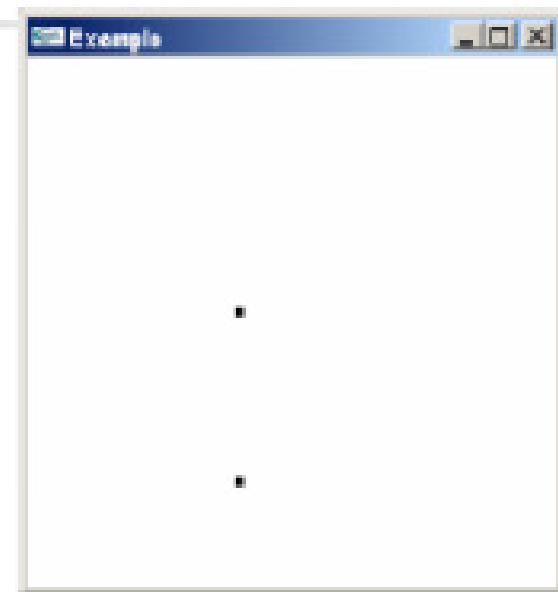




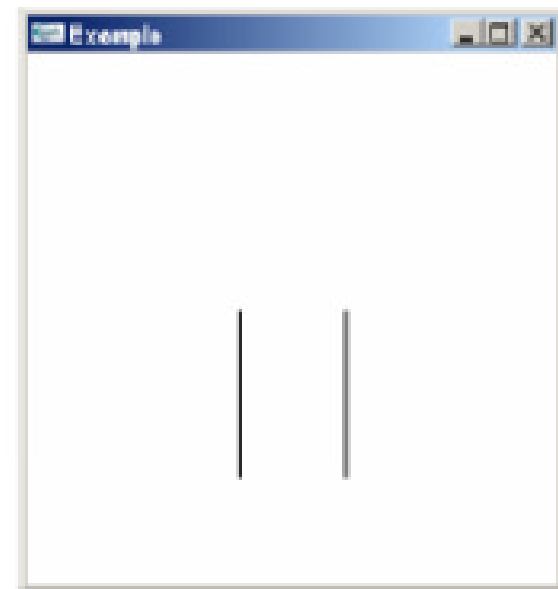
Linhas, Pontos e Polígonos

■ Exemplo:

```
glBegin(GL_POINTS);  
    glColor3f(0.0f, 0.0f, 0.0f);  
    glVertex2i(100, 50);  
    glVertex2i(100, 130);  
glEnd();
```



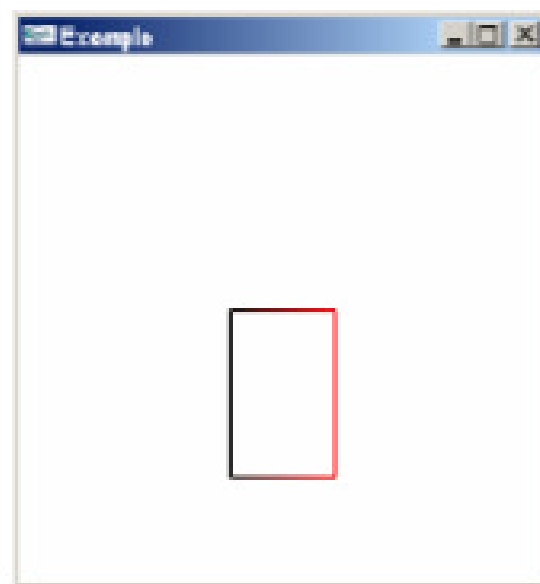
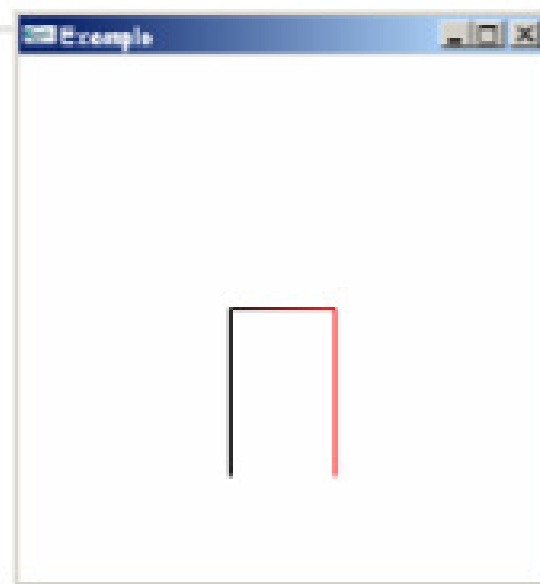
```
glBegin(GL_LINES);  
    glColor3f(0.0f, 0.0f, 0.0f);  
    glVertex2i(100, 50);  
    glVertex2i(100, 130);  
    glVertex2i(150, 130);  
    glVertex2i(150, 50);  
glEnd();
```

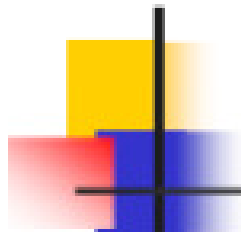


Linhas, Pontos e Polígonos

■ Exemplo:

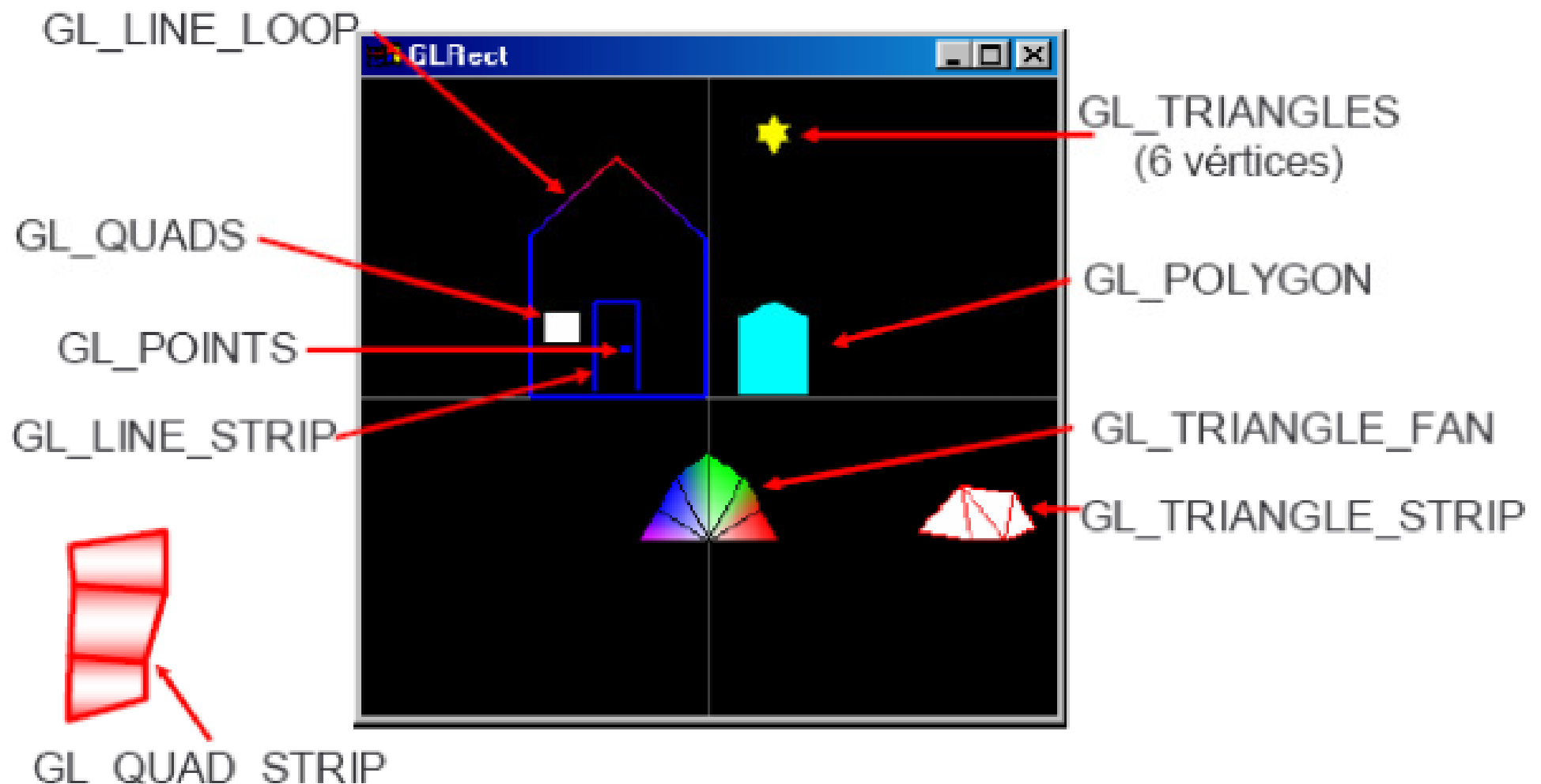
```
glBegin(GL_LINE_STRIP);  
    glColor3f(0.0f, 0.0f, 0.0f);  
    glVertex2i(100, 50);  
    glVertex2i(100, 130);  
    glColor3f(1.0f, 0.0f, 0.0f);  
    glVertex2i(150, 130);  
    glVertex2i(150, 50);  
  
glEnd();  
  
glBegin(GL_LINE_LOOP);  
    glColor3f(0.0f, 0.0f, 0.0f);  
    glVertex2i(100, 50);  
    glVertex2i(100, 130);  
    glColor3f(1.0f, 0.0f, 0.0f);  
    glVertex2i(150, 130);  
    glVertex2i(150, 50);  
  
glEnd();
```





Linhas, Pontos e Polígonos

■ Outros exemplos



Referências:

- ◆ WOO, Mason; NEIDER, Jackie; DAVIS, Tom; SHREINER, Dave. **OpenGL Programming Guide: the official guide to learning OpenGL, version 1.2.3rd ed.** Reading, Massachusetts: Addison Wesley, 1999. 730 p.
- ◆ WRIGHT, Richard S. Jr.; SWEET, Michael. **OpenGL SuperBible.** 2nd ed. Indianapolis, Indiana: Waite Group Press, 2000. 696 p.

