Computação Gráfica

Prof. Rodrigo Martins
rodrigo.martins@francomontoro.com.br

Esse material foi cedido pelo Prof. Jorge Cavalcanti da UNIVASF
(UNIVERSIDADE FEDERAL DO VALE DO SÃO FRANCISCO)

Introdução

- OpenGL (*Open Graphics Library* – www.opengl.org) é uma biblioteca de rotinas gráficas e de modelagem bidimensional e tridimensional, extremamente portável e rápida;
- Também definida como Interface para Hardware Gráfico;
- É uma API (*Aplication Program Interface*) para aceleração da programação de dispositivos gráficos;
- Não é uma linguagem de programação.
- Aplicação OpenGL é escrita em alguma linguagem de programação e utiliza uma ou mais bibliotecas OpenGL;
- Existem bibliotecas específicas para cada linguagem de programação.

Características

- Rapidez e portabilidade;
- Existem bibliotecas para várias plataformas (Windows, Linux, Macintosh, etc.);
- Algumas linguagens que implementam aplicações gráficas utilizando OpenGL: C, C++, Java, C# e Python;
- Além de primitivas gráficas, dá suporte a iluminação e sombreamento, mapeamento de textura, transparência, animação, etc.
- É reconhecida e aceita como um padrão API para desenvolvimento de aplicações gráficas 3D em tempo real.
- Possui aproximadamente 250 comandos e funções.
- Várias bibliotecas disponíveis:

https://www.opengl.org/resources/libraries/

Configuração do Ambiente

- OpenGL Utility Toolkit
  - Sistema de Janelas independente de plataforma para desenvolvimento de aplicações OpenGL
  - Possui funções para:
    - Criar/Destruir janelas
    - Tratar entradas de teclado, mouse e joysticks
- Baseado em funções de *callback* para tratamento de eventos
- API simples, não possuindo recursos diretos para criação de GUI's
- Independente do sistema de janelas nativo
- Programação orientada a eventos

Bibliotecas

- GLU
  - É instalada junto com a OpenGL;
  - Contém funções que encapsulam comandos de baixo nível;
  - Nome das funções utilizam prefixo glu;
  - Possui funções para modelagem, como superfícies quádricas, curvas e superfícies.

Bibliotecas

- GLUT (https://www.opengl.org/resources/libraries/glut/)
  - Realiza tarefas como:
    - Criar e gerenciar as janelas da aplicação OpenGL;
    - Criar e gerenciar menus nas janelas;
    - Desenhar objetos padrões como esferas, cilindros e paralelepípedos;
    - Desenhar textos;
    - Tratar eventos de teclado, mouse e joystick.

Sintaxe de Comando

- Todos os nomes das funções seguem um padrão para facilitar a utilização. Esses nomes indicam:
  - Qual a biblioteca que a função faz parte
  - Quantos e que tipos de argumentos a função tem.
- Convenção adotada:

  <prefixbiblioteca><cmdraiz><contargopc><tpargopc>

Ex.: glColor3f

Prefixo que representa a biblioteca gl

Comando raiz

Sufixo que significa que a função tem 3 valores float como parâmetro

Sintaxe de Comando

- O contador do número de argumentos e tipo dos argumentos permitem a criação de várias funções com o mesmo objetivo
  - glColor3i(GLint red, GLint green, GLint blue);
  - glColor3f(GLdouble red, GLdouble green, GLdouble blue);
- Outras variações da função glColor recebem quatro argumentos;
  - glColor4f(GLfloat red, GLfloat green, GLfloat blue, GLfloat alpha);

Sintaxe de Comando

- Os nomes das funções das outras bibliotecas, como GLU e GLUT, possuem apenas o prefixo da biblioteca e o comando raiz, com os parâmetros definidos.
  - void gluOrtho2D(GLdouble left, GLdouble right, GLdouble width, GLdouble height);
  - void glutSolidCube(Gldouble size);

Tipos de dados

- Para portar o código OpenGL de uma plataforma para outra mais facilmente, foram definidos tipos de dados próprios para OpenGL.
- Estes tipos de dados são mapeados os tipos de dados C comuns, que também podem ser utilizados.
  - Porém, os vários compiladores e ambientes possuem regras diferentes para determinar o tamanho das variáveis C.
- Usando os tipos OpenGL é possível, então, "isolar" o código das aplicações destas alterações.

Tipos de dados

Tipo de dado OpenGL	Represent. interna	Tipo dado C equivalente	Sufixo
GLbyte	8-bit integer	signed char	b
GLshort	16-bit integer	short	s
GLint, GLsizei	32-bit integer	int ou long	i
GLfloat, GLclampf	32-bit floating-point	float	f
GLdouble, GLclampd	64-bit floating-point	double	d
GLubyte, GLboolean	8-bit unsigned integer	unsigned char	ub
GLushort	16-bit unsigned integer	unsigned short	us
GLuint, GLenum, GLbitfield	32-bit unsigned integer	unsigned long / unsigned int	ui

- As constantes usam uma notação semelhante às funções.
  - Usa-se o prefixo da biblioteca
  - Depois usa-se "_" entre as palavras do nome da variável, em maiúsculas

GLUT_RIGHT_BUTTON
GLUT_LEFT_BUTTON

Máquina de Estados

- Dispositivo ou sistema que guarda o estado de um ou mais elementos em um momento específico;
- OpenGL é uma máquina de estados composta de muitas variáveis de estado. Estas variáveis armazenam, por exemplo: estilo da linha, espessura da linha, propriedades do material dos objetos;
- Pode-se usar uma função para alterar uma variável de estado mais de uma vez durante a execução de um programa;
  - As variáveis de estado podem ser habilitadas ou desabilitadas através das funções: *void glEnable() e void glDisable*(). Veja o trecho de código a seguir:

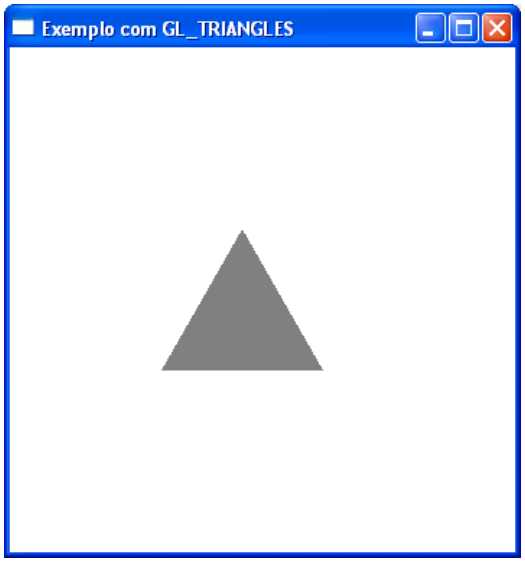
Máquina de Estados

```
int estilo_linha;
...
//Habilita alterar o estilo de uma linha -
glEnable(GL_LINE_STIPPLE);  //GL_LINE_STIPPLE  var. de
  estado
...
// retorna 1 (verdadeiro)
estilo_linha = glIsEnabled(GL_LINE_STIPPLE);
...
//Desabilita alterar o estilo de uma linha
glDisable(GL_LINE_STIPPLE);
…
// retorna 0 (falso)
estilo_linha = glIsEnabled(GL_LINE_STIPPLE);
…
```

Exemplo adaptado de COHEN & MANSSOUR [2006]

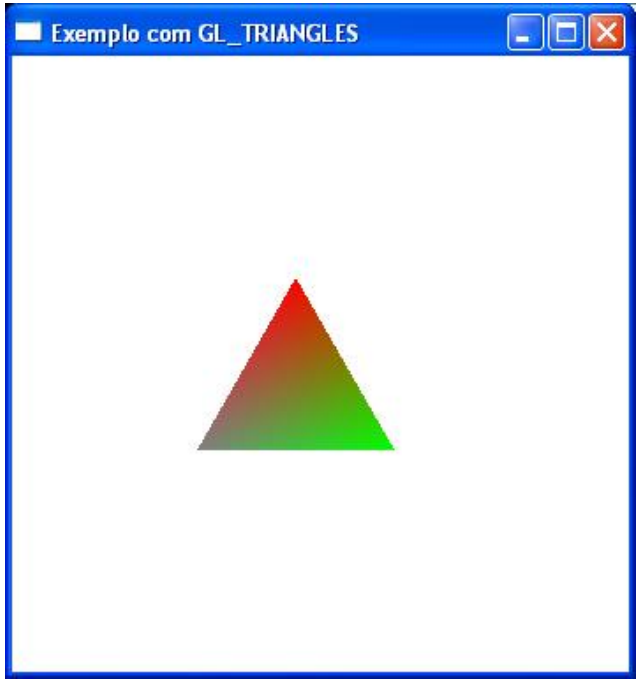
Máquina de Estados

```
...
// Altera a cor do desenho para cinza
   glColor3f(0.5f, 0.5f, 0.5f);
...
   // Desenha um triângulo na cor corrente
   glBegin(GL_TRIANGLES);
      glVertex2f(-20.0f,-14.0f);
      glVertex2f(-4.0f, 14.0f);
      glVertex2f( 12.0f,-14.0f);
   glEnd();
```



Máquina de Estados

```
...
// Altera a cor do desenho para cinza
    glColor3f(0.5f, 0.5f, 0.5f);
...
    // Desenha um triângulo
    glBegin(GL_TRIANGLES);
        //Vértice na cor corrente
        glVertex2f(-20.0f,-14.0f);
        // setando a cor Vermelha
        glColor3f(1.0f, 0.0f, 0.0f);
        glVertex2f(-4.0f, 14.0f);
        // setando a cor Verde
        glColor3f(0.0f, 1.0f, 0.0f);
        glVertex2f( 12.0f,-14.0f);
    glEnd();
```



Estrutura de uma aplicação interativa

- Configura e abre uma janela
- Inicializa OpenGl
  - Limpa a tela
  - Define matriz de projeção
- Registra as funçoes Callback de entrada
  - Desenho
  - Alterações do tamanho da janela
  - Entrada de dados via teclado ou mouse
- Processamento de eventos

Primeiro programa

*main.cpp

```
1
2   #ifdef __APPLE__
3   #include <GLUT/glut.h>
4   #else
5   #include <GL/glut.h>
6   #endif
7
8   #include <stdlib.h>
9
10  void Inicializa(void){
11      // Define a cor de fundo da janela de visualização como azul
12      glClearColor(0.0f, 0.0f, 1.0f, 1.0f);
13  }
14
15  // Função callback chamada para fazer o desenho
16  void Desenha(void){
17      //Limpa a janela de visualização com a cor de fundo especificada
18      glClear(GL_COLOR_BUFFER_BIT);
19
20      //Executa os comandos OpenGL para renderização.
21      glFlush();
22  }
```

Primeiro programa

```
23
24  int main(int argc, char *argv[])
25  {
26      glutInit(&argc, argv);
27      glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
28      glutCreateWindow("Primeiro Programa");
29      glutDisplayFunc(Desenha);
30      Inicializa();
31      glutMainLoop();
32
33      return EXIT_SUCCESS;
34  }
```

Exemplo de um programa

- O arquivo glut.h contém os protótipos das funções utilizadas pelo programa.
- Ele também inclui os headers gl.h e glu.h que definem, respectivamente, as bibliotecas de funções OpenGL e GLU.
- O header windows.h é requerido por todas as aplicações windows, mas a sua inclusão é opcional porque a versão WIN32 da GLUT já inclui o windows.h na glut.h.
  - Entretanto, se o objetivo é criar um código portável, é um bom hábito incluir este arquivo.

Funções de Inicialização da GLUT

- **glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB)-** avisa a GLUT que tipo de modo de exibição deve ser usado quando a janela é criada.
  - Neste caso os flags indicam a criação de uma janela single-buffered (GLUT_SINGLE) com o modo de cores RGBA (GLUT_RGB).
  - O primeiro significa que todos os comandos de desenho são feitos na janela de exibição.
  - Uma alternativa é uma janela *double-buffered*, onde os comandos de desenho são executados para criar uma cena fora da tela para depois rapidamente colocá-la na view.
  - Este método é geralmente utilizado para produzir efeitos de animação.
  - O modo de cores RGBA significa que as cores são especificadas através do fornecimento de intensidades dos componentes Red, Green e Blue separadas. A é o indicador de transparência.

Funções de Inicialização da GLUT

- **glutInitDisplayMode**
  - Especifica o modelo de cor, a utilização de *single* ou *double-buffer* e quais buffers OpenGL serão utilizados na janela GLUT que será aberta.

    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
  - Outras constantes utilizadas:
  - **GLUT_DOUBLE, GLUT_DEPTH**

Buffer é uma área de memória onde é armazenada uma matriz de valores, que correspondem aos pixels de uma imagem

Funções de Inicialização da GLUT

- **glutCreateWindow**
  - Cria uma janela GLUT que permite a execução de comando OpenGL, recebendo o título desta como parâmetro.
  - glutCreateWindow("Primeiro Programa");
- **glutInitWindowPosition**
  - Define a posição inicial na tela, do canto superior esquerdo da janela.

glutInitWindowPosition(int x, int y);

x – Coordenada x do canto superior esquerdo da janela.

y – Coordenada y do canto superior esquerdo da janela.

- **glutInitWindowSize**
  - Define o tamanho inicial da janela GLUT que será criada.

glutInitWindowSize(int largura, int altura);

glutInitWindowSize(400,400);

Funções de Inicialização da GLUT

- **glutDisplayFunc**
  - Define que uma função será responsável por redesenhar a janela OpenGL sempre que necessário.
  - É nesta função que deve-se colocar as chamadas de funções de *rendering* OpenGL.

```
glutDisplayFunc(Desenha);
```

- **glutMainLoop**
  - Inicia o processamento de eventos da GLUT, não retornando para o programa do usuário.

```
glutMainLoop();
```

  - A partir do momento que esta função é chamada, o controle do programa passa à GLUT, que inicia o gerenciamento dos eventos.
  - Nenhum código após essa chamada será executado.

Primeiro programa

- **Inicializa();** não é uma função OpenGL nem GLUT, é apenas uma convenção utilizada no livro no qual este material está baseado.
  - Nesta função são feitas as inicializações OpenGL que devem ser executadas antes da exibição do desenho (rendering). Muitos estados OpenGL devem ser determinados somente uma vez e não a cada vez que o rendering é realizado.
- **glClearColor(0.0, 0.0, 1.0, 1.0);** é a função que determina a cor utilizada para limpar a janela.
  - Seu protótipo é: void glClearColor(GLclampf red, GLclampf green, GLclampf blue, GLclampf alfa);
  - GLclampf O componente alfa é usado para efeitos especiais, tal como transparência. O intervalo para cada componente red, green, blue é de 0 a 1.
- **glClear(GL_COLOR_BUFFER_BIT);** "limpa" o buffer de pixels, removendo eventuais resíduos de outra aplicação gráfica.

Primeiro programa

Acrescentar na função inicializa:

```
// Define a janela de visualização 2D
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0.0,10.0,0.0,10.0);
```

Acrescentar na função desenha entre:

```
glClear(GL_COLOR_BUFFER_BIT)
...
glFlush();
```

```
// Define a cor de desenho: vermelho
glColor3f(1.0,0.0,0.0);
// Desenha um triângulo no centro da janela
glBegin(GL_TRIANGLES);
        glVertex3f(2.0, 2.0, 0);
        glVertex3f(6.0, 2.0, 0);
        glVertex3f(4.0, 6.0, 0);
glEnd();
```

Primeiro programa

```
// Função callback chamada para gerenciar eventos de teclas
void Teclado (unsigned char key, int x, int y)
{
    if (key == 27)
        exit(0);
}
```

Acrescentar no programa principal (main):

```
//Chamada da Função de entrada de dados via teclado
glutKeyboardFunc (Teclado);
```

Primeiro programa

- Faça as seguintes alterações no arquivo "PrimeiroPrograma":
  - Altere a cor do fundo para amarelo;
  - Faça com que o programa seja encerrado ao pressionar a tecla q;
  - Altere o programa de modo que a janela de visualização tenha os seguintes valores:

    X: 400
    Y: 280

  - Altere o tamanho da janela GLUT para 300 X 200.

Segundo programa

main.cpp

```
1   #ifdef __APPLE__
2   #include <GLUT/glut.h>
3   #else
4   #include <GL/glut.h>
5   #endif
6
7   #include <stdlib.h>
8   #include <windows.h>
9
10  // Um programa OpenGL simples que desenha um
11  // quadrado em  uma janela GLUT.
12  // Este código está baseado no GLRect.c, exemplo
13  // disponível no livro "OpenGL SuperBible",
14  // 2nd Edition, de Richard S. e Wright Jr.
15
16
17  // Função callback chamada para fazer o desenho
18  void Desenha(void)
19  {
20      glMatrixMode(GL_MODELVIEW);
21      glLoadIdentity();
22
23      // Limpa a janela de visualização com a cor de fundo especificada
24      glClear(GL_COLOR_BUFFER_BIT);
25
26      // Especifica que a cor corrente é vermelha
27      //          R     G     B
28      glColor3f(1.0f, 0.0f, 0.0f);
```

Segundo programa

```
29
30          // Desenha um quadrado preenchido com a cor corrente
31          glBegin(GL_QUADS);
32          glVertex2i(100,150);
33          glVertex2i(100,100);
34
35         // Especifica que a cor corrente é azul
36         glColor3f(0.0f, 0.0f, 1.0f);
37         glVertex2i(150,100);
38         glVertex2i(150,150);
39          glEnd();
40
41          // Executa os comandos OpenGL
42          glFlush();
43  }
44
45  // Inicializa parâmetros de rendering
46  void Inicializa (void)
47  {
48          // Define a cor de fundo da janela de visualização como preta
49          glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
50  }
```

Segundo programa

```
51
52  // Função callback chamada quando o tamanho da janela é alterado
53  void AlteraTamanhoJanela(GLsizei w, GLsizei h)
54  {
55      // Evita a divisao por zero
56      if(h == 0) h = 1;
57
58      // Especifica as dimensões da Viewport
59      glViewport(0, 0, w, h);
60
61      // Inicializa o sistema de coordenadas
62      glMatrixMode(GL_PROJECTION);
63      glLoadIdentity();
64
65      // Estabelece a janela de seleção (left, right, bottom, top)
66      if (w <= h)
67          gluOrtho2D (0.0f, 250.0f, 0.0f, 250.0f*h/w);
68      else
69          gluOrtho2D (0.0f, 250.0f*w/h, 0.0f, 250.0f);
70  }
71
72
```

Segundo programa

```
73  int main(int argc, char *argv[])
74  {
75      glutInit(&argc, argv);
76      glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
77      glutInitWindowSize(400,350);
78      glutInitWindowPosition(10,10);
79      glutCreateWindow("Quadrado");
80      glutDisplayFunc(Desenha);
81      glutReshapeFunc(AlteraTamanhoJanela);
82      Inicializa();
83      glutMainLoop();
84
85
86      return EXIT_SUCCESS;
87  }
```

Referências desta aula

- AZEVEDO, Eduardo; CONCI, Aura. 2007. Computação Gráfica: Teoria e Prática. Elsevier, Vol. 2, 2007.

- Aula montada com base no material do Prof. Jorge Cavalcanti - UNIVASF.