

Aula 3: Gráficos com OpenGL

Apresentação

O uso de Application Programming Interfaces (API) mais elaboradas é um desafio, pois, em geral, exige a consulta de sua documentação e a busca por exemplos já desenvolvidos para que possamos chegar a uma compreensão de como executar o que necessitamos. Um bom exemplo é a API OpenGL, que veremos nesta aula.

Com muitas funções e um modo particular de operar com uma máquina de estados, a API é um bom teste para desenvolvermos o entendimento básico do acesso ao monitor de vídeo do sistema e aos princípios de animação de objetos gráficos.

Objetivos

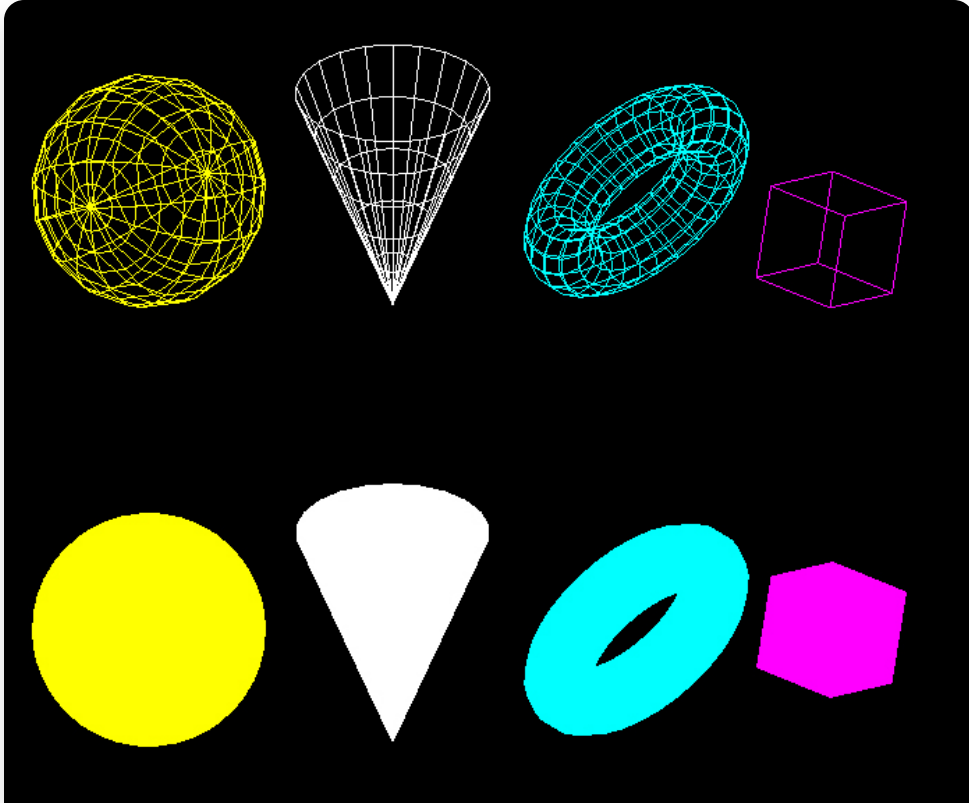
- Descrever o uso da biblioteca OpenGL em C;
- Desenvolver programas com interface e recursos visuais;
- Definir os conceitos de animação de objetos gráficos.

API OpenGL

OpenGL, abreviação de Open Graphics Library, é uma API projetada para renderizar gráficos 2D e 3D. Ele fornece um conjunto comum de comandos que podem ser usados para gerenciar gráficos em diferentes aplicativos e em várias plataformas.

Ao usar o OpenGL, um desenvolvedor pode usar o mesmo código para renderizar gráficos em um Mac, PC ou dispositivo móvel. Quase todos os sistemas operacionais e dispositivos de hardware modernos oferecem suporte ao OpenGL, tornando-o uma escolha fácil para o desenvolvimento de gráficos.

Além disso, muitas placas de vídeo e Graphics Processing Units (GPU) – ou Unidade de Processamento Gráfico – integradas são otimizadas para o OpenGL, permitindo que eles processem comandos do OpenGL com mais eficiência do que outras bibliotecas gráficas.



 Exemplos de comandos. (Fonte: Software OpenGL).

Exemplos de comandos do OpenGL incluem desenhar polígonos, atribuir cores a formas, aplicar texturas a polígonos (mapeamento de textura), aumentar e diminuir o zoom, transformar polígonos e girar objetos. O OpenGL também é usado para gerenciar efeitos de iluminação, como fontes de luz, sombreado e sombras. Também pode criar efeitos como neblina, que podem ser aplicados a um único objeto ou a uma cena inteira.

O OpenGL:

- 1 É comumente associado a videogames, por causa de seu amplo uso em jogos 3D. Ele fornece aos desenvolvedores uma maneira fácil de criar jogos de plataforma cruzada ou portar um jogo de uma plataforma para outra.
- 2 Também é empregado como biblioteca de gráficos para muitos aplicativos CAD, como AutoCAD e Blender. Até a Apple usa o OpenGL como base das bibliotecas de gráficos macOS Core Animation, Core Image e Quartz Extreme.
- 3 Foi originalmente desenvolvido e lançado pela Silicon Graphics (SGI) em 1992. A versão inicial foi aprovada por um conselho de revisão de arquitetura, que incluía Microsoft, IBM, DEC e Intel. Em 2006, a SGI transferiu o desenvolvimento e a manutenção do OpenGL para o The Khronos Group.

Saiba mais

Antes do OpenGL e de outras APIs gráficas, como a DirectX, os jogos tinham que usar instruções especiais, dependendo da placa gráfica de que se dispunha. Ao comprar um novo jogo, você tinha que verificar cuidadosamente se seu cartão era suportado.

A CPU (que é onde o programa que você escreve é executado) não pode falar diretamente com a tela ou o monitor. Em vez disso, você envia os dados que deseja desenhar para a Graphics Processing Unit (GPU) – ou Unidade de Processamento Gráfico.

Assim, a GPU desenha os dados e APIs gráficas, como OpenGL, permitem que programas em execução na CPU enviem dados para a GPU e personalizem como esta os desenha.

Utilização da OpenGL

Embora a API OpenGL seja uma biblioteca poderosa e versátil, com cerca de 250 funções, funcionando em muitas plataformas diferentes para a construção de ambientes, os princípios de programação seguem os de qualquer outra biblioteca em C. As tarefas ligadas ao sistema operacional não são tratadas pela OpenGL, o que inclui funções para arquivos e janelas, sendo esta uma característica importante para a portabilidade entre sistemas.

Em razão da ampla adoção do OpenGL em todos os tipos de sistemas, como Linux e Windows, além de iOS e Android, este não é mais considerado uma única API. O OpenGL evoluiu em uma família de APIs, também com aplicações em sistemas móveis e web, permitindo que os desenvolvedores de aplicativos escrevam e implantem aplicativos gráficos em uma ampla variedade de plataformas e sistemas operacionais.

Bibliotecas de utilitários GLUT e OpenGL

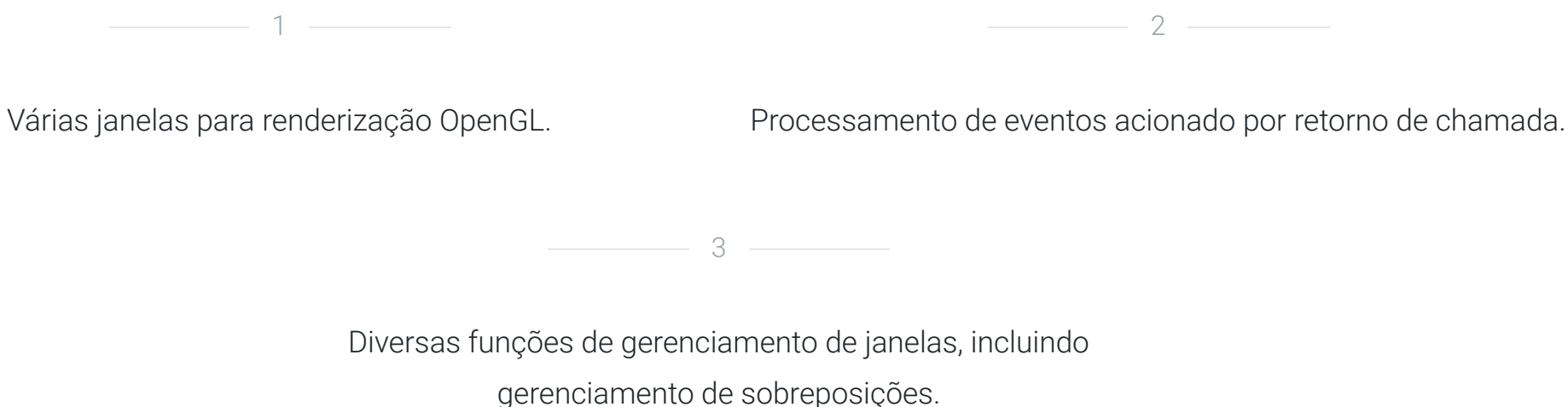
Existem inúmeras bibliotecas de sistemas e interfaces de janelas disponíveis para o OpenGL. OpenGL Utility Toolkit (GLUT) é um kit de ferramentas independente do sistema de janelas para escrever programas OpenGL e também para o tratamento de eventos de dispositivos de entrada (mouse e teclado). Ele implementa uma simples API para o OpenGL. O GLUT facilita consideravelmente o aprendizado e a exploração do OpenGL.

Comentário

O GLUT, portanto, fornece código de utilitário para interagir com o gerenciador de janelas de seu sistema operacional, para que você possa (mais facilmente) solicitar um recurso, que é basicamente um contexto gráfico e uma fila de eventos de entrada.

Você usa o OpenGL para renderizar primitivas no contexto gráfico que não tem conhecimento de mais nada da janela, como entrada do usuário, decoração da janela, eventos da janela (como redimensionar ou minimizar) ou mesmo o próprio sistema operacional. O OpenGL só sabe lidar com gráficos 3D: buffers, viewport, matrizes de transformação, polígonos, texturas, modelos de luz, sombreado etc.

O kit de ferramentas GLUT suporta as seguintes funcionalidades:



O GLUT simplifica a implementação de programas usando a renderização OpenGL. A API GLUT requer muito poucas rotinas para exibir uma cena gráfica renderizada usando o OpenGL. A API GLUT (como a API OpenGL) é estável. A maior parte do estado GLUT inicial é definido e é razoável para programas simples

Por esse motivo, o GLUT não retorna nenhum identificador de sistema de janela nativo, ponteiros ou outras estruturas de dados. Dependências mais sutis do sistema de janelas, como a dependência de fontes, são evitadas pelo GLUT; em vez disso, o GLUT fornece seu próprio conjunto, limitado, de fontes.

As coordenadas da janela e da tela GLUT são expressas em pixels. O canto superior esquerdo da tela ou uma janela é (0,0). As coordenadas X aumentam na direção para a direita, e as coordenadas Y aumentam na direção descendente.

O GLUT suporta vários retornos de chamada para responder a eventos. Existem três tipos de retornos de chamada:

Janela

Menu

Global

Os retornos de chamada da janela indicam quando exibir novamente ou remodelar uma janela, quando a visibilidade da janela é alterada e quando a entrada está disponível para a janela.

OpenGL no Dev C++

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

Na implementação do OpenGL no Dev C++, o sistema operacional Windows já fornece as DLLs `opengl32.dll` e `glu32.dll` (GLU – biblioteca de utilitários OpenGL), necessárias para execução de programas OpenGL. Como o Dev C++ dá suporte para programação com OpenGL, a pasta `Include\GL` contém os arquivos `gl.h`, `glaux.h` e `glu.h`, e a pasta `Lib` contém os arquivos `opengl32.def`, `glaux.def` e `glu32.def`.

Para simplificar a tarefa de programação, utiliza-se também a biblioteca GLUT, que pode ser instalada seguindo os passos:

Passo 1

Baixar a biblioteca GLUT no [link](#).



Passo 2

Extrair os arquivos para a pasta C:/GLUT.

Passo 3

Os quatro arquivos (figura 1) poderiam ser transferidos para as pastas LIB e Include do compilador, mas colocando-os em uma pasta única é possível informar ao compilador para usar essa pasta ao buscar por arquivos de biblioteca e cabeçalhos (headers .h). Isso é feito, no Dev C++, em Ferramentas > Opções do Compilador, aba Diretórios (figura 2), clicando no ícone com a seta, indicando a pasta C:/GLUT e optando por adicionar, tanto na aba Bibliotecas, como na aba C Includes.



Passo 4

Copiar o arquivo glut32.dll para a pasta C:\Windows\system, C:\Windows\system32, ou, ainda, para a pasta onde o projeto será criado.

Passo 5


Para testar, criar um projeto no Dev C++ com as opções Console Application e C Project. Acrescentar em Projeto > Opções do Projeto, na aba Parâmetros, na coluna Linker, as instruções lglut32, lglu32 e lopengl32 (figura 3)



Passo 6

No arquivo principal do projeto, acrescentar o código a seguir (aparecerá uma janela com fundo preto e um quadrado branco e o título "Ola Mundo" na janela)
Veja o código .

As principais funções da GLUT utilizadas são:

 Clique nos botões para ver as informações.

[glutInitDisplayMode\(GLUT_SINGLE | GLUT_RGB\).](#)



Refere-se ao modo de exibição inicial usado ao criar janelas, subjanelas e sobreposições de nível superior para determinar o modo de exibição do OpenGL para a janela ou a sobreposição a ser criada.

[glutDisplayFunc\(display\).](#)



Chama a função para redesenhar a janela.

[glutMainLoop\(\):](#)



Inicia a máquina de estados do programa gráfico.

[glutCreateWindow](#)



Cria uma janela de nível superior. O nome será fornecido ao sistema de janelas como o nome da janela. A intenção é que o sistema de janelas rotule a janela com o nome. Implicitamente, a janela atual é definida como a janela recém-criada. Cada janela criada tem um contexto OpenGL associado exclusivo. Alterações de estado no contexto OpenGL associado a uma janela podem ser feitas imediatamente depois da criação desta.

Os arquivos de cabeçalho para GLUT devem ser incluídos nos programas GLUT com a seguinte diretiva de inclusão:

#include <glut.h>

Como o processamento de geometria é realizado pelo sombreador de vértices, esse programa simples exige que quatro posições de vértice separadas (glVertex2f) sejam enviadas todas as vezes em que queremos exibir o retângulo.

Máquina de estado

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

O OpenGL é, por si só, uma grande máquina de estado:

Uma coleção de variáveis que definem como o OpenGL deve operar. O estado do OpenGL é geralmente chamado de contexto do OpenGL. Ao usar o OpenGL, geralmente mudamos seu estado, definindo algumas opções, manipulando alguns buffers e, depois, processando com uso do contexto atual.

Exemplo

Sempre que dizemos ao OpenGL que agora queremos desenhar linhas em vez de triângulos, por exemplo, mudamos o estado do OpenGL alterando alguma variável de contexto que define como o OpenGL deve desenhar. Assim que alteramos o estado, dizendo ao OpenGL que ele deve desenhar linhas, os próximos comandos de desenho passam a desenhar linhas em vez de triângulos.

Ao trabalhar no OpenGL, encontraremos várias funções de alteração de estado que modificam o contexto e várias funções de uso do estado que executam algumas operações com base no estado atual do OpenGL. Desde que você tenha em mente que o OpenGL é basicamente uma grande máquina de estado, a maior parte de sua funcionalidade fará mais sentido.

Desenhando elementos gráficos com OpenGL

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

Vamos agora inserir um exemplo que mostra como é realizado um desenho no OpenGL, apresentando também como são feitos o movimento e o redimensionamento da janela.

```
#include <windows.h>
#include <glut.h>

void Desenha(void)
{
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();

    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1.0f, 0.0f, 0.0f);

    glBegin(GL_QUADS); //Quadrado de cor azul
    glVertex2i(100,150);
    glVertex2i(100,100);

    glColor3f(0.0f, 0.0f, 1.0f);
    glVertex2i(150,100);
    glVertex2i(150,150);
    glEnd();
    glFlush();
}
```

```
void Inicializa (void)
{
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
}

void MudaTamanhoJanela(GLsizei w, GLsizei h)
{
    if(h == 0) h = 1;

    glViewport(0, 0, w, h);

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();


    if (w <= h)
        gluOrtho2D (0.0f, 250.0f, 0.0f, 250.0f*h/w);
    else
        gluOrtho2D (0.0f, 250.0f*w/h, 0.0f, 250.0f);
}

int main(void)
{
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(500,350);
    glutInitWindowPosition(50,30);
    glutCreateWindow("Quadrado");
    glutDisplayFunc(Desenha);
    glutReshapeFunc(MudaTamanhoJanela);
    Inicializa();
    glutMainLoop();
}
```

Assim, temos:

 Execução do programa. (Fonte: Software OpenGL).

As novas funções usadas são:

 Clique nos botões para ver as informações.

[glutInitWindowSize\(500,350\).](#)



Define o tamanho, em pixels, da janela.

[glutInitWindowPosition\(50,30\).](#)



Define a localização da janela, a posição a 50 pixels do topo e a 30 pixels da esquerda da janela do Windows;

[glutReshapeFunc\(MudaTamanhoJanela\).](#)



Define a função para mudar o tamanho da janela. A função "MudaTamanhoJanela" é chamada para recomençar o sistema de coordenadas da janela.

[glColor3f\(1.0f, 0.0f, 0.0f\).](#)



Estabelece a cor usada para realizar o desenho.

[glBegin\(GL_QUADS\); até glEnd\(\).](#)



Nesse conjunto de comandos, realiza-se o desenho de um quadrado. A API OpenGL verifica as coordenadas, alterando-as para a posição atual na função já vista, a "MudaTamanhoJanela".

A chamada de função que mais se destaca é o `glViewport`. Uma viewport define a área desenhada pelo OpenGL. Qualquer ponto que fique fora da janela de visualização não é desenhado. A razão pela qual isso é chamado da viewport é porque é uma janela para a cena, olhando apenas uma pequena área de uma cena possivelmente muito maior.

É importante notificar a OpenGL sobre quaisquer alterações na janela de visualização, para não desenhar pixels fora da tela, resultando em cálculos desperdiçados que seriam mais bem gastos em outros lugares.

A função `glViewport` usa quatro parâmetros, a saber:

Coordenada X

Coordenada Y

Largura

Altura

As coordenadas X e Y correspondem ao canto inferior esquerdo da janela de visualização. Em nosso caso, o canto inferior esquerdo da janela (0, 0). Essa função também permite que você desene para uma área muito específica em uma janela menor no interior da janela. Por exemplo, isso pode ser útil se você precisar incorporar gráficos OpenGL em uma interface do usuário.

A viewport é especificada com a utilização das funções descritas a seguir:

1

glViewport(0, 0, w, h)

Armazena os parâmetros de largura e altura da janela para que esta seja redimensionada.

2

**gluOrtho2D
(0.0f, 250.0f*w/h, 0.0f, 250.0f)**

Estabelece que a projeção ortográfica, a projeção 2D, será a usada para a imagem 2D que se apresenta na janela de seleção.

3

**glMatrixMode
(GL_PROJECTION)**

Indica que as próximas alterações irão mudar como o observador verá a janela.

4

glLoadIdentity()

Estabelece que a matriz atual será inicializada juntamente com a matriz identidade.

Atividade

1. Quais das afirmativas são verdadeiras (V) e quais são falsas (F) quanto às vantagens da API OpenGL?

- a) O OpenGL é portátil.
- b) O OpenGL permite animações e temporização.
- c) É uma API de plataforma cruzada (cross-platform).

2. Vimos que, no estudo de uma API mais complexa, como a OpenGL, observar programas prontos como exemplo é essencial para o aprendizado. O programa abaixo desenha uma linha diagonal preta. Diga quais linhas alterar para desenhar uma linha horizontal vermelha.

```
#include <glut.h>
#include <stdlib.h>

void init(void);
void display(void);
void keyboard(unsigned char key, int x, int y);

int main(int argc, char** argv){
    glutInit(&argc, argv); //Inicializa o GLUT
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (256, 256); //Especifica as dimensões da janela
    glutInitWindowPosition (100, 100); //Especifica aonde a janela aparece na tela
    glutCreateWindow ("Desenhando uma linha"); //Cria a janela
    init();

    glutDisplayFunc(display); //Função que será redeseenhada pelo GLUT
    //glutKeyboardFunc(keyboard); //Funções de teclado
    glutMainLoop(); //Mostra todas as janelas criadas
    return 0;
}

//Definição de cada função

void init(void){
    glClearColor(1.0, 1.0, 1.0, 1.0); //Cor de fundo
    glOrtho (0, 256, 0, 256, -1 ,1); //Modo de projeção ortogonal
}

void display(void){
    glClear(GL_COLOR_BUFFER_BIT); //Limpa a janela
    glColor3f (0.0, 0.0, 0.0); //Cor da linha
    glBegin(GL_LINES);
    glVertex2i(40,200); glVertex2i(200,10); //Coordenadas inicial e final da linha
    glEnd();
    glFlush();
}
```

3. Altere o programa apresentado na figura 4 de modo que o quadrado se mova horizontalmente na tela.

4. As afirmações A e/ou B podem ser definidas como desvantagem do OpenGL em relação a outras APIs gráficas?
- A) Por natureza, o OpenGL é restrito a um único sistema operacional.
- B) Para chamadas de desenho e alterações de estado, o OpenGL tem uma sobrecarga de CPU maior do que as outras APIs gráficas.
5. Nos exemplos dados nesta aula, nós usamos o kit de ferramentas utilitárias GLUT, que é um sistema baseado em eventos. Como foi realizada a interação do OpenGL com o GLUT?

Notas

(figura 1)

(figura2)

(figura 3)

código

```
#include <glut.h>
void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_POLYGON);
    glVertex2f(-0.3, -0.3);
    glVertex2f(-0.3, 0.3);
    glVertex2f(0.3, 0.3);
    glVertex2f(0.3, -0.3);
    glEnd();
    glFlush();
}
```

```
}  
int main(void) {  
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);  
    glutCreateWindow("Ola Mundo");  
    glutDisplayFunc(display);  
    glutMainLoop();  
}
```

Referências

MANSSOUR, Isabel Harb. **Introdução à OpenGL**. Disponível em: <https://www.inf.pucrs.br/~manssour/OpenGL/Utilizacao.html>. Acesso em: 26 dez. 2019.

SELLERS, Graham; WRIGHT, Richard S.; HAEMEL, Nicholas. **OpenGL Superbible**: Comprehensive Tutorial and Reference. 7th ed. Addison-Wesley Professional, 2015.

VRIES, Joey de. **Learn OpenGL**. Disponível em: <https://learnopengl.com/book/offline%20learnopengl.pdf>. Acesso em: 26 dez. 2019.

Próxima aula

- Bibliotecas em C para captura de eventos de teclado e mouse, incluindo acesso a registros usando C;
- Entrada e saída por console e arquivo;
- Acesso à porta serial.

Explore mais

O aprendizado aprofundado de OpenGL é algo que demanda muito tempo, em razão da quantidade de funções que estão disponíveis na API. Este primeiro contato com OpenGL tem a intenção de mostrar a utilização de uma API grande e poderosa que atua no sistema operacional no nível mais baixo, como é a intenção desta disciplina. Este material ajuda no aprofundamento do estudo da API:

- [Introdução à computação gráfica com OpenGL](https://www.ft.unicamp.br/~magic/opengl/index2006.html); <<https://www.ft.unicamp.br/~magic/opengl/index2006.html>>
- [Introdução à OpenGL](https://www.inf.pucrs.br/~manssour/OpenGL/Programando3D.html). <<https://www.inf.pucrs.br/~manssour/OpenGL/Programando3D.html>>