O que é OpenGL ?

- Open Graphics Library
- "*Uma interface de software com o hardware gráfico*"
- Uma camada de abstração entre o programa de aplicação e o hardware gráfico
- Construída na forma de API em C (Application Programmer's Interface)
- Independente do sistema operacional

Características Principais

- Projetada para aplicações gráficas interativas 2D e 3D
- Descendente de GL (Graphics Library - SGI)
- Controlada pelo ARB (Architecture Review Board)
  - 3DLabs, Compaq/Digital, Evans&Sutherland, HP, IBM, Intel, InterGraph, Microsoft, Nvidia, Silicon Graphics

- Permite criar programas interativos que produzem imagens coloridas de objetos em movimento
- MESA (www.mesa3d.org) FREE!

Aspectos Técnicos

- Primitivas são vértices e imagens
- O desenho das primitivas é afetado pelo estado atual das variáveis de controle
  - escala, rotação, translação, cor, iluminação, textura, largura de linhas, etc.
- As variáveis de controle têm valores *default*
- Não gerencia eventos de controle (mouse, exibição, teclado, etc)

GLU (Graphics Utility Library)

- Conjunto de rotinas utilizadas freqüentemente
  - Construídas a partir de comandos OpenGL
- Rotinas para:
  - Manipulação de projeções
  - Desenho de superfícies quádricas
  - NURBS
  - Manipulação de superfícies poligonais

Sintaxe dos comandos

`glVertex3fv`

- Comandos iniciam com `gl`
- `3` -> número de componentes
- `f` -> tipo de dado (float, *d*ouble float, *s*igned short integer, s*i*gned integer)
- `v`-> formato escalar ou array (vector)

Desenhando

- Primitivas são delimitadas por `glBegin()` e `glEnd()`
- `glVertex*()`especifica o vértice
- Atributos de um vértice
  - `glColor*()`, `glNormal*()`, `glMaterial*()`
- Tipos de primitivas
  - `GL_POINTS`, `GL_LINES`, `GL_QUADS`, `GL_POLYGON`, `GL_TRIANGLES`, `etc.`

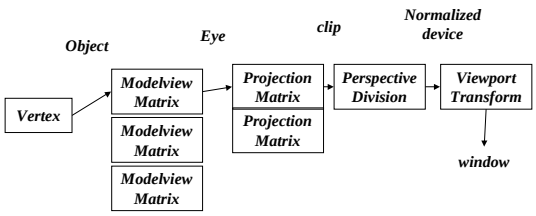
Exemplo

- Desenhar um triângulo verde e plano (z=0)

```
glBegin(GL_TRIANGLE);
    glColor3f(0,1,0);
    glNormal3f(0,0,1);
    glVertex3f(0,0,0);
    glVertex3f(1,0,0);
    glVertex3f(0,1,0);
glEnd();
```

- Remoção de elementos ocultos feita por *z-buffer*

Pipeline de Visualização



Transformações

- Matrizes de Projeção
  - `glFrustum(left,right,bottom,top,near,far)`
  - `glOrtho(left,right,bottom,top,near,far)`
  - `gluPerspective(fovy,aspect,zNear,zFar)`
  - `gluOrtho2D(left,right,bottom,top)`
  - `gluLookAt(eyex, eyey, eyez, centerx, centery, centerz, upx, upy, upz)`
- Coordenadas de Tela
  - `glViewport(x, y,width, height)`

Transformações

- Modelagem
  - `glTranslate{fd}(x, y, z)`
  - `glRotate{fd) (angle, x, y, z )`
  - `glScale{fd}(x,y,z)`
- Propriedades Materiais `glMaterial*()`
  - ambiente, difuso, especular
- Fontes de Luz `glLight*()`
  - cor, posição, atenuação, etc

Mais informação

- Sites
  - www.sgi.com/software/opengl
  - www.opengl.org
- Livros
  - OpenGL Programming Guide (Woo, Neider, Davis - Addison-Wesley)
  - Interactive Computer Graphics: A Top-Down Approach with OpenGL (E. Angel - Addison-Wesley)