# ESZI017 – Fundamentos de Processamento Gráfico

*Laboratório 02:* **Introdução à Estrutura de OpenGL**

## 1. Introdução

Esta aula prática tem como objetivo apresentar um ambiente para programação em linguagem C++ e a biblioteca **OpenGL**, devido à praticidade proporcionada e à grande difusão desta ferramenta no meio acadêmico.

Este roteiro deverá ser executado no sistema **Ubuntu-linux** instalado no laboratório didático. Instruções são apresentadas e exemplos que o aluno deve executar no laboratório, e verificar os resultados obtidos. Ao final do roteiro são propostos alguns exercicios para entregar que o aluno deverá realizar individualmente.

## 2. Instalação da biblioteca FreeGLUT no Ubuntu-linux

**(a) A instalação da biblioteca FreeGLUT no Ubuntu-linux deverá ser realizado conforme o roteiro da aula anterior.**

**(b) A Compilaçào e execução dos programas OpenGL, deverão ser realizados conforme o roteiro da aula anterior. Não será utilizado IDE.**

Exemplo de compilação:

```
gcc -o test lesson5.cpp -lglut -lGL -lGLU
```

Exemplo de execução:

```
./test
```

## 3. Escrevendo um programa OpenGL

***i*) Abrir um novo programa**

Para cada novo programa, crie uma nova sub-pasta na sua pasta principal. O nome da sub-pasta deverá ser "lab02-*NN*", onde *NN* é o número sequencial dos programas do roteiro. Portanto, neste primeiro programa *NN* = 01.

-Para escrever o código, poderá ser utilizado o editor de programa "**Geany**".

-Num terminal linux, vá ao diretório criado e digite o nome do editor "`>geany`".

-Neste editor, crie um novo arquivo, e salve no diretório usando o nome "lab02-*NN*", com a extensão "cpp". Exemplo: "lab02-01.cpp" .

***ii*) Adicionar código fonte e salvar o programa:**

```
/*
 *  Meu Primeiro Programa: xxx.cpp
 *  <NomeCompleto>, RA <RA>, data DD/MM/AAAA
 */
#include <GL/gl.h>
#include <GL/glu.h>
#include <GL/glut.h>
#include <stdlib.h>

// Função callback chamada para fazer o desenho
void Desenha(void)
{
    //Limpa a janela de visualização com a cor de fundo especificada
    glClear(GL_COLOR_BUFFER_BIT);

    //Executa os comandos OpenGL
    glFlush();
}

// Inicializa parâmetros de rendering
void Inicializa (void)
{
    // Define a cor de fundo da janela de visualização como preta
    glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
}

// Programa Principal
int main(int argc, char *argv[])
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutCreateWindow("Meu Primeiro Programa");
    glutDisplayFunc(Desenha);
    Inicializa();
    glutMainLoop();
    return 0;
}
```

***iii*) Compilar e Executar o projeto no terminal linux, conforme o exemplo inicial.**

## 4. Atividades: Desenhando Polígonos com OpenGL

Na atividade desta parte desenharemos polígonos num plano, conforme já explanado nas aulas teóricas. O objetivo é realizar o traçado das primitivas de desenho do OpenGL, permitindo ao aluno elaborar posteriormente modelos de objetos com este recurso.

Para tanto iremos usar o programa “hello” dos exemplos do RedBook como base, e modifica-lo para compor os desenhos.

a) Abra no editor Geany, o programa **`hello`**, que está no arquivo hello.c. Salve-o como “hello.cpp”. Estude passo a passo cada comando de código.

b) Pelo terminal, compile e execute o programa, verificando seu funcionamento.

Dentro deste programa, observe que o trecho de código responsável pelo desenho do polígono é o seguinte:

```
gl.glBegin(GL.GL_POLYGON);
  gl.glVertex3f(0.25f, 0.25f, 0.0f);
  gl.glVertex3f(0.75f, 0.25f, 0.0f);
  gl.glVertex3f(0.75f, 0.75f, 0.0f);
  gl.glVertex3f(0.25f, 0.75f, 0.0f);
gl.glEnd();
```

c) Altere este trecho de código para que seja apresentado na janela gráfica, um primitiva de cada vez, conforme os tipos de primitivas apresentadas na Figura 1. Para cada caso, salve o programa e a imagem resultante.

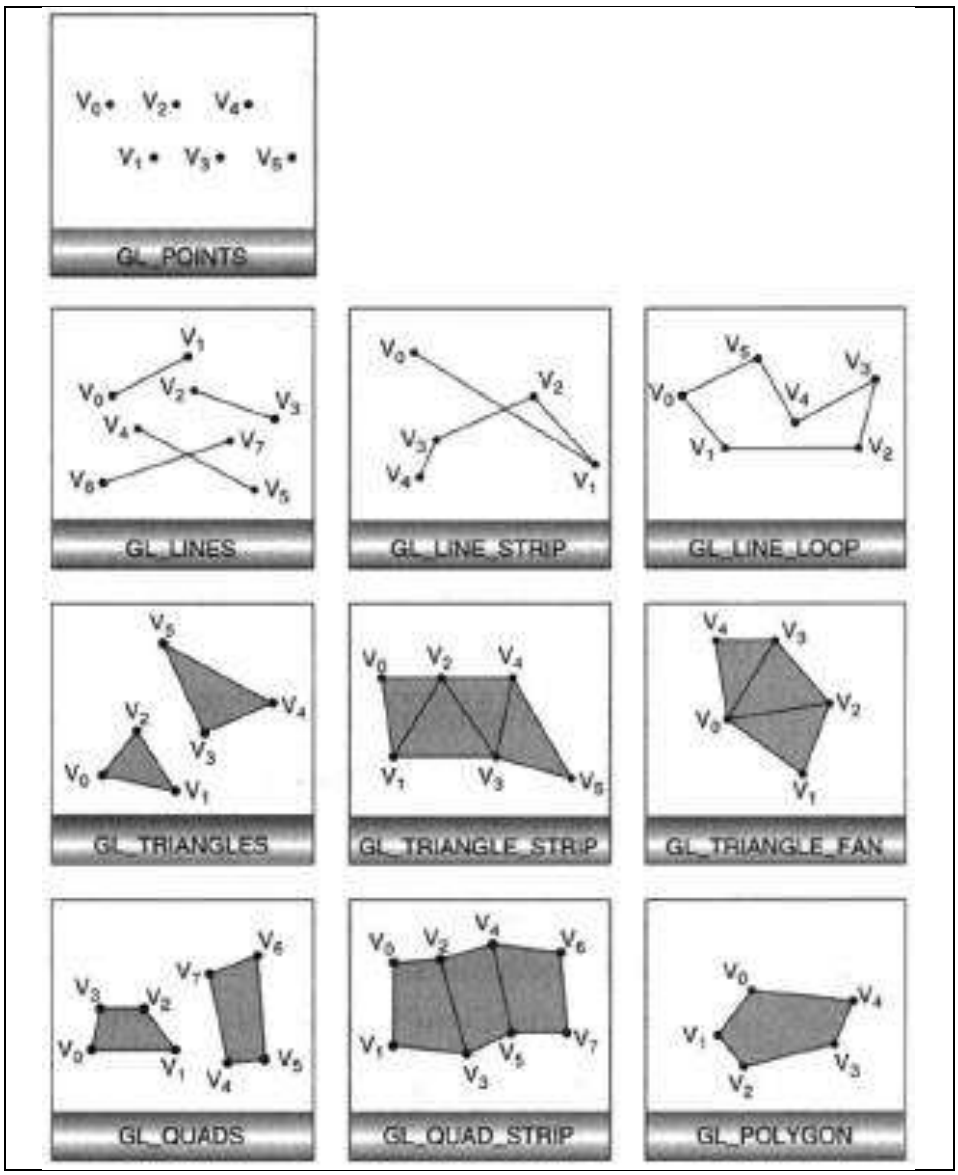


Figura 1: Ilustração das primitivas de desenho gráfico do OpenGL.

d) Em seguida, avalie o comando de cores, como o mostrado no trecho de código abaixo. Execute-o e apresente os resultados.

```
glBegin( GL.GL_TRIANGLES );
   glColor3f( 1.f, 0.f, 0.f );
   glVertex3f( 0.2f, 0.2f, 0.f );
   glColor3f( 0.f, 1.f, 0.f );
   glVertex3f( 0.8f, 0.2f, 0.f );
   glColor3f( 0.f, 0.f, 1.f );
   glVertex3f( 0.2f, 0.8f, 0.f );
glEnd();
```

OBS.: veja a definição de algumas cores em OpenGL:

| | |
|---|---|
| glColor3f(0.0, 0.0, 0.0) | black |
| glColor3f(1.0, 0.0, 0.0) | red |
| glColor3f(0.0, 1.0, 0.0) | green |
| glColor3f(1.0, 1.0, 0.0) | yellow |
| glColor3f(0.0, 0.0, 1.0) | blue |
| glColor3f(1.0, 0.0, 1.0) | magenta |
| glColor3f(0.0, 1.0, 1.0) | cyan |
| glColor3f(1.0, 1.0, 1.0) | white |

e) Para se traçar um círculo, pode-se fazer uma aproximação, pois não há uma primitiva em OpenGL para isso. O trecho de código abaixo calcula alguns pontos do círculo para serem ligados com uma linha. Execute-o e apresente os resultados.

```
GLdouble PI = 3.1415926535897;
GLint circle_points = 100;

glBegin(GL.GL_LINE_LOOP);
for (i = 0; i < circle_points; i++) {
   angle = 2*PI*i/circle_points;
   glVertex2f(cos(angle), sin(angle));
}
glEnd();
```

## 5. Exercícios para Entregar

A resolução deverá ser em programa C++, e a entrega através do TIDIA.
**Importante:** em todos os arquivos de código faça um cabeçalho (com comentários) incluindo seu nome completo, RA, data do programa, nome do programa, e exemplo de chamada do programa no prompt do linux.

1) Baseado nos programas desenvolvidos nesta aula prática, elabore um programa que apresente uma figura geométrica que se aproxime à da figura abaixo.

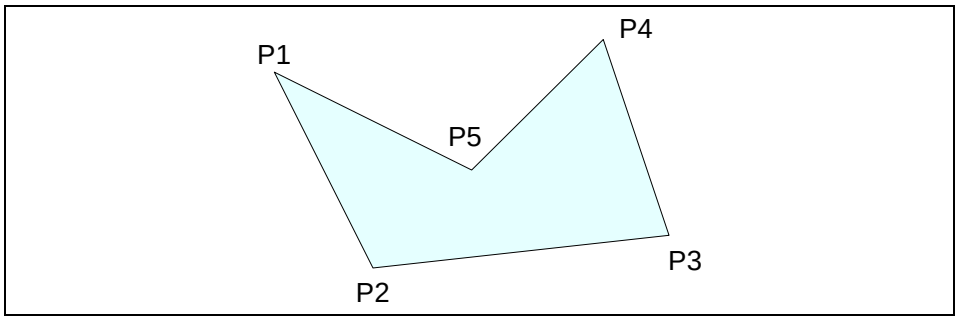

Figura 2: Figura do exercício 1 para entregar.

2) Baseado nos programas desenvolvidos nesta aula prática, elabore um programa que apresente uma figura geométrica que se aproxime à da figura abaixo.

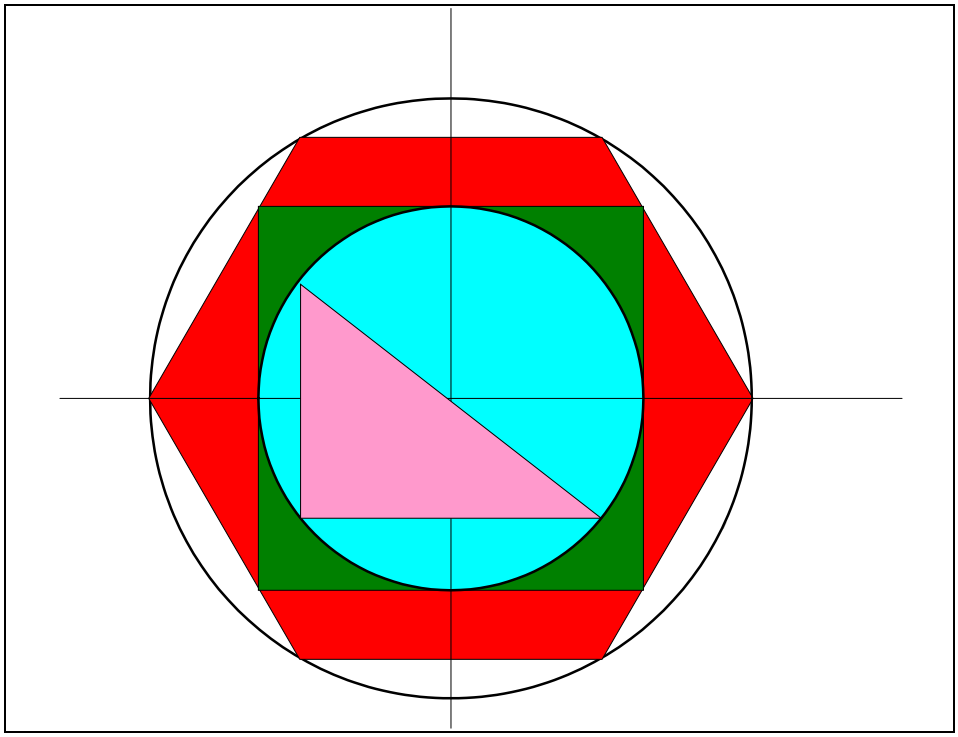
Figura 3: Figura do exercício 2 para entregar.

## 6. Referências

[1] Dave Shreiner; the Khronos OpenGL ARB Working Group. **OpenGL programming guide : the official guide to learning OpenGL, versions 3.0 and 3.1.** 7th ed. Addison-Wesley, 2009.

## 7. Relatório

Envie o relatório na forma de arquivo eletrônico **HTML**, e os **programas C++**, com cabeçalho (comentário) incluindo seu nome completo, RA, data do programa, nome do programa, e exemplo de chamada do programa no prompt do linux.
Elaborar um relatório contendo:
- Os procedimentos detalhados executados no laboratório.
- Imagens obtidas. Tabelas de análises comparativas de resultados.
- Análise e conclusões.
- Exercícios para entregar