UNIVERSIDADE ESTADUAL DE SANTA CRUZ (UESC)

Criada pela Lei 6.344, de 05.12.1991,
e reorganizada pela Lei 6.898, de 18.08.1995 e
pela Lei 7.176, de 10.09.1997

CET115 – Processamento Digital de Imagens

OpenGl

Prof. Dra. Vânia Cordeiro da Silva
Departamento de Ciências Exatas e Tecnológicas
Universidade Estadual de Santa Cruz (UESC)
vania(at)uesc(dot)br

OpenGl vs DirectX

- DirectX: coleção de APIs para programação de jogos (multimídia) do SO MS-Windows
  - DirectX e OpenGL: interfaces mais populares
  - Baixado gratuitamente do site da Microsoft
    - Até quando?
  - Programação 3D com som
  - Críticas:
    - Não portável
    - Pouco suporte
    - Xbox não suporta OpenGL

Sistemas Gráficos

- OpenGL (GL – Graphics Library)**:** sistema gráfico mais utilizado no mundo hoje em dia
- Primeira versão: 92, relativamente novo
- Baseado na biblioteca gráfica GL das workstation IRIS da Silicon Graphics, Inc.:
  - Algoritmos cuidadosamente otimizados

OpenGl

- Atualmente um consorcio livre de indústrias é responsável pelo gerenciamento da evolução:
  - www.opengl.org
  - 3DLabs, Compaq/Digital, Evans&Sutherland, HP, IBM, Intel, InterGraph, Microsoft, Nvidia, Silicon Graphics
- Implementação livre: código fonte disponível
  - Freeglut, Mesa3D ou MesaGL
  - Gratuita, código aberto, roda no linux e windows

OpenGl

- Suporta iluminação, sombreamento, animação, oclusão, mapeamento de textura, transparência, etc.
- Permite criar objetos gráficos com qualidade de modo mais rápido, inclui recursos avançados de animação, tratamento de imagens e texturas

OpenGl

- Interface de software para dispositivos de HW
  - + ou – 250 comandos
- Biblioteca gráfica de modelagem e exibição 3D, rápida e portável para vários S.O.
  - Independente do sistema operacional e do HW
- Uma camada de abstração entre o programa de aplicação e o hardware gráfico

OpenGl

- Ponte entre processo de modelagem geométrica situada no nível de abstração mais elevado e rotinas de exibição e de processamento de imagens implementadas em dispositivos (hw) e S.O. específicos
- Construída na forma de API em C (Application Programmer’s Interface)

OpenGl

- Funcionamento semelhante a biblioteca C
- Todas as rotinas são implementadas em C, tornando fácil sua utilização em qualquer programa escrito em C ou C++.
- Funções: possui mesmos nomes, parâmetros e efeito de exibição em todos os S.O.

O que OpenGl não faz

- Portabilidade: não trabalha diretamente com o sistema de janelas do ambiente operacional
- Não possui comandos de alto nível para construir objetos de 3 dimensões (para isso existem bibliotecas como GLU – *OpenGL Utility Library* que é padrão em qualquer implementação de OpenGL)
- Bibliotecas auxiliares: GLU e GLUT
- GLU: instalada junto com OpenGL

O que OpenGl não faz

- As especificações do OpenGL não descrevem as interações entre este e o sistema de janelas utilizado (Windows, X Window etc)
  - Cada ambiente tem o seu
- Não processa nenhuma entrada de usuário (teclado, mouse, etc.)

O que OpenGl não faz

- GLUT (OpenGL Utility Toolkit): funções para gerenciamento de janelas, interação com o usuário ou arquivos de entrada/saída
  - Menus ou suporte a joystick
  - Não é de domínio público, mas é gratuita
- SDL (Simple Directmedia Layer): Biblioteca multimídia voltada para jogos

O que OpenGl não faz

- Programas mais complexos: toolkit mais completo
  - FLKT (Flash Light Toolkit) e wxWidgets(GTK): elaboração de interfaces
  - Open Inventor: cenários 3D
  - Etc: http://www.opengl.org/resources/libraries/higherlevel/

Instalação com GLUT

- 3 bibliotecas: OpenGL, GLU e GLUT
  - Específicos para cada compilador: os nomes são os mesmos, mas os conteúdos não
- OpenGL: pré-instalada no windows a partir do 98 e no linux, na maioria das distribuições

Instalação

- gcc/gnu
- Visual C++
- Borland C++
- Delphi
- Java
- Visual basic
- Dev C++
- CodeBlocks

Instalação

- Windows/Visual C + +:
  - Instalar a biblioteca GLUT
  - Criar um projeto para linkar as bibliotecas necessárias com o programa
  - Incluir os headers adequados (<gl/gl.h> e <gl/glu.h>, ou <gl/glut.h>)

Instalação

- Dev C++:
  - Compilador: http://www.bloodshed.net/devcpp.html
  - O ambiente *Windows* já deve conter as DLLs necessárias: opengl32.dll e glu32.dll

Instalação

- Dev C++:
  - Tem suporte para programação com OpenGL
    - Pasta GL: gl.h, glaux.h e glu.h
    - Pasta LIB: opengl32.def, glaux.def e glu32.def
  - Falta GLUT:
    - Ferramentas/Atualizações/WebUpdate
    - Selecione servidor devpaks.org e clique em Check for updates
    - Selecione o grupo OpenGL, marque glut e clique em Download selected

Instalação

- Dev C++: alternativamente
  - Download da página
  - Mova o arquivo *glut.h* para a pasta GL
  - Mova os arquivos *glut32.def* e *libglut.a* para a pasta Lib
  - Mova o arquivo *glut32.dll*

Instalação

- Dev C++:
  - Necessário criar projeto (menu arquivo)
  - Clique na aba Multimedia e glut
  - Escolha linguagem (C++) e digite nome
  - Compile e execute programa-exemplo
  - Remova arquivo e selecione novo arquivo

Instalação

- Code::Blocks: ambiente de desenvolvimento (IDE) recomendado para uso do OpenGL com as linguagens C/C++ (MinGW e gcc/g++), multiplataforma (windows e linux) e código aberto
  - www.codeblocks.org
  - Traz consigo pacotes do OpenGL, exceto GLUT
  - Windows: instalação semelhante ao DEV
    - .h na pasta GL, .lib na pasta lib...
  - Procure /usr/include/GL ou /usr/X11R6/include/GL
    - Já deve conter gl.h e glu.h

Instalação

- Linux:
  - Linux: instalar glut, build-essential e libxxf86vm-dev
    - sudo apt-get install codeblocks freeglut3-dev build-essential libxxf86vm-dev
  - Linux 64bits:
    - sudo ln -s /usr/lib/x86_64-linux-gnu/libglut.a /usr/lib/libglut.a
    - sudo ln -s /usr/lib/x86_64-linux-gnu/libglut.so /usr/lib/libglut.so
    - sudo ln -s /usr/lib/x86_64-linux-gnu/libglut.so.3 /usr/lib/libglut.so.3
    - sudo ln -s /usr/lib/x86_64-linux-gnu/libglut.so.3.9.0 /usr/lib/libglut.so.3.9.0

Instalação

- Code::Blocks: já traz um assistente para criação de um projeto OpenGL com janela gráfica GLUT
  - Settings-> Compiler and debugger-> Selected compiler, selecione GNU GCC Compiler
  - Arquivo-> New-> Project-> GLUT project-> GO-> Next-> coloque nome e defina a localização de seu projeto-> Next-> /usr no campo de localização-> Next 2x

Instalação

- Compilando em terminal :
  - gcc <arquivo.c> -o nome_saída –lglut –lGL –lGLU –lm
- Arquivo de LOT (BATH):

```
#!/bin/sh

gcc $1 -o $2 -lglut -lGL -lGLU -lm
```

  - Necessário: chmod 777 NOME.sh
    - Duvidas: man chmod

Tipos de Dados

- Código-fonte portável: são definidos tipos de dados próprios para OpenGL
- Mapeados dos tipos de dados C comuns,

  que também podem ser utilizados

Tipos de Dados

Tipo de dado OpenGL	Representação interna	Tipo de dado C equivalente
GLbyte	8-bit integer	signed char
GLshort	16-bit integer	short
GLint, GLsizei	32-bit integer	int ou long
GLfloat, GLclampf	32-bit floating-point	float
GLdouble, GLclampd	64-bit floating-point	double
GLubyte, GLboolean	8-bit unsigned integer	unsigned char
GLushort	16-bit unsigned integer	unsigned short
GLuint, GLenum, GLbitfield	32-bit unsigned integer	unsigned long ou unsigned int

Convenção para Nomes de Funções

- Indica de qual biblioteca a função faz parte e, freqüentemente, quantos e que tipos de argumentos a função tem

  <PrefixoBiblioteca> <ComandoRaiz>

  <ContadorArgumentosOpcional>

  <TipoArgumentosOpcional>

Convenção para Nomes de Funções

- glColor3f(GLfloat R, GLfloat G, GLfloat B);
  - gl: prefixo que induca biblioteca gl
  - Color: comando raiz que indica objetivo da função
  - 3: indica o número de argumentos que a função possui
  - f: argumentos em ponto flutuante
- Várias funções, mesmo objetivo: glVertex2i() e glVertex3f()

Convenção para Nomes de Funções

- Funções sem parâmetros:
  - glFlush(void)
  - glutMainLoop(void)
  - glLoadIdentity(void)
- Número de parâmetros não variam:
  - void glTranslatef(GLfloat x, GLfloat y, GLfloat z)

Operação

- Programas com OpenGL podem se tornar bastante complicados diante do volume de operações que podem ser realizadas
- Entretanto, a estrutura básica de vários programas é relativamente simples
  - Os mains costumam ter a mesma estrutura

Operação

- Seqüência de operações:
  - Abra uma janela gráfica
  - Prepare OpenGL para desenhar na janela
  - Defina o sistema de coordenadas e o estado inicial do OpenGL: limites
  - Loop

Operação

- Seqüência de operações (cont.):
  - Loop:
    - Trate os eventos de mouse e teclado
    - Mude a cena com base nos eventos ocorridos
    - Redesenhe a cena com OpenGL
  - Funções de callback: quem chama a função não é o programador, e sim a GLUT
    - Devem ser especificadas no início do programa

Primitivas Gráficas de Desenho

- Elementos básicos que compõem um desenho: como pontos, segmentos de retas, círculos, polígonos ...
- Definidas em um sistema de coordenadas: vértices
- Objetos e cenas complexos consistem em combinação das primitivas gráficas de desenho

Primitivas Gráficas de Desenho

- Em Java3D são 7: pontos, linhas, triângulos e quadrados simples e com variações
- Em Directx são 3: pontos, linhas e triângulos todos simples
- Em OpenGL são 10...

Primitivas de Desenho

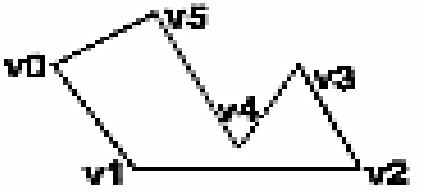
- GL_POINTS: desenha pontos
- GL_LINES: segmentos de linha
- GL_LINE_STRIP: segmentos conectados
- GL_LINE_LOOP: liga o último ao primeiro ponto
- GL_QUADS: quadriláteros
- GL_QUAD_STRIP: quadriláteros conectados
- GL_POLYGON: polígonos convexos
- GL_TRIANGLES: desenha triângulos
- GL_TRIANGLE_STRIP: triângulos conectados
- GL_TRIANGLE_FAN: a partir de um ponto central

Primitivas de Desenho

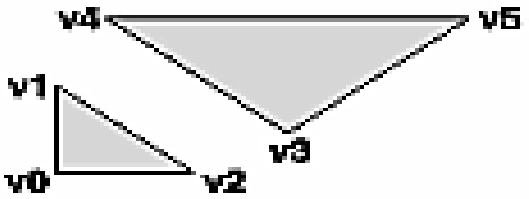
GL_POINTS

GL_LINES

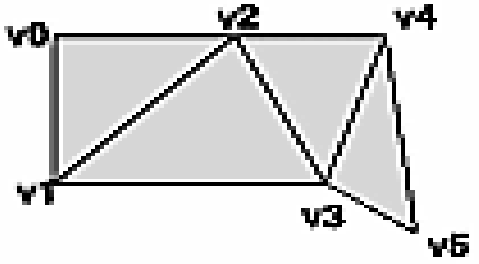
GL_LINE_STRIP



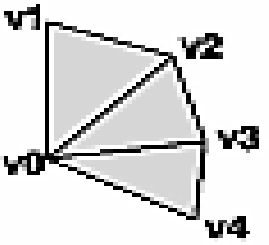
GL_LINE_LOOP



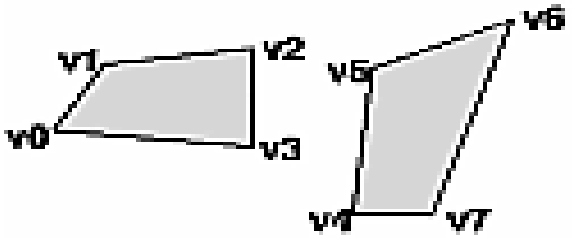
GL_TRIANGLES



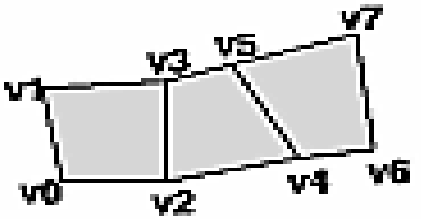
GL_TRIANGLE_STRIP



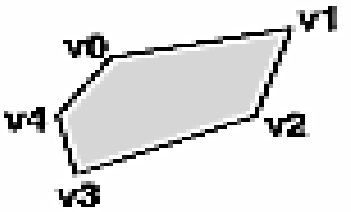
GL_TRIANGLE_FAN



GL_QUADS



GL_QUAD_STRIP



GL_POLYGON

Atividade 01

- Estudar o código pdi.c e modificá-lo para abrir e apresentar qualquer imagem da família PNM no modo texto: P1, P2 e P3
  - Identificar a extensão automáticamente
  - Ignorar comentários quando presentes
  - Normalizar a profundidade caso seja necessário
- Data da entrega: 25/10