

Librerías Gráficas Introducción a OpenGL

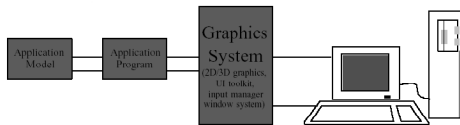


Introducción

- Sistema gráfico :
 - Modelos + Visualización
- Visualización:
 - Uso de hardware específico (2D o 3D)
 - Implementación a través de librerías

Definición de librería gráfica

- Software que genera imágenes en base a unos modelos matemáticos y unos patrones de iluminación, texturas, etc.



Librerías, ficheros, sintaxis

- Origen
 - IGL-Plot 10 (Tektronix)
 - Starbase (Hewlett Packard)
 - Iris GL Library (SGI)
- Distinguir la naturaleza de otros sistemas gráficos
 - VRML, X3D (Lenguajes de descripción)
 - DirectX-Direct3D
 - Java 3D
 - Open Inventor
 - Performer, Fahrenheit
 - Motores

Objetivos de las librerías gráficas

- Independencia del hardware (tanto dispositivos de entrada como de salida).
- Independencia de la aplicación (la librería es accedida a través de un interface único (al menos para cada lenguaje de programación) para cualquier aplicación).

Tipos de librerías gráficas

- Direct Rendering and gfx packages:
 - OpenGL, Direct3D, GKS, PHIGS, PEX, GKS, etc...
- Scene-graph based
 - OpenGL Performer, Open Inventor, OpenGL Optimizer, PHIGS+, etc...
- Toolkits
 - World Toolkit, AVANGO, Game Engines, etc...

- Gestión imágenes 3D ...
 - “Bajo nivel”
 - Tareas
 - Gestión “en serie” de los elementos de la escena
 - Elementos de la escena
 - » Primitivas gráficas
 - » “Atributos” (“edición imágenes”)
 - Variables de estado ↘ Generación imagen
 - Sistemas
 - OpenGL
 - Direct 3D
 - Java 3D
 - “Alto nivel” ...

- (... Gestión imágenes 3D) ...
 - “Alto nivel”
 - Tareas
 - Gestión global de los elementos de la escena
 - Árbol escena
 - Carga/descarga de memoria
 - Gestión elementos no visibles
 - Elección del modelo geométrico: Nivel detalle, Textura
 - Elección de la técnica de presentación (rendering)
 - Sistemas
 - Inventor
 - Performer
 - (Fahrenheit)
 - Hewlett Packard

DirectX Componentes

- DirectDraw
- DirectSound
- DirectPlay
- Direct3D
-

DirectX - Introducción

- ¿Qué es DirectX?
 - Conjunto de API’s que permite a los desarrolladores de contenido interactivo (imagen, video, sonido...) acceder a características de hardware especializado sin tener que escribir código específico de hardware

DirectX - Introducción

- Componentes incluidos en DirectX
 - Nos permiten desarrollar aplicaciones de alto rendimiento y en tiempo real
 - API **Direct Play**
 - API **Direct Input**
 - API **Direct Sound**
 - API **Direct Draw**
 - API **Direct 3D**

DirectX - Introducción

- Objetivos de DirectX
 - Desarrollo de Aplicaciones Windows de alto rendimiento
 - Tarjetas aceleradoras
 - Plug’n Play
 - Servicios de comunicaciones construidos bajo Windows
 - Recursos instalados en el sistema
 - Utilización del nuevo hardware implementado

DirectX - Introducción

- DirectX & COM (Component Object Model)
 - **Objeto**: caja negra que representa el hardware y requiere comunicación con las aplicaciones a través de una interface.
 - **Método**: comandos enviados y recibidos por el objeto a través de la interface COM
- Ej.: Método **Get DisplayMode** es enviado a través de la interface **IDirectDraw2** para tomar el valor de la actual resolución de la pantalla mediante el objeto **Direct Draw**

DirectX - DirectDraw

- Se encarga del manejo de la memoria de vídeo
- Proporciona herramientas para
 - Manipulación de múltiples buffers de vídeo
 - Acceso directo a la memoria de vídeo
 - Page flipping
 - Back Buffering
 - Manejo de la paleta gráfica
 - Clipping

DirectX - DirectDraw

- Tipos de objetos
 - **IDirect Draw**
 - **IDirect DrawSurface**
 - **IDirect DrawPalette**
 - **IDirect DrawClipper**
 - **IDirect DrawVideoPort**

DirectX - DirectDraw

- Conceptos Gráficos y Técnicos
 - Bitmaps
 - Superficies de dibujo (buffers)
 - Page Flipping y Back Buffering
 - Rectángulos
 - Sprites
 - Niveles cooperativos
 - Modos de vídeo
 - Buffers
 - Overlays
 - Clippers
 - Video Ports

DirectX - DirectSound

- Componente de Audio de DirectX:
 - Mezclado de canales de audio
 - Aceleración hardware
 - Acceso directo al dispositivo de sonido
 - Captura de Audio

DirectX - DirectSound

- Interfaces COM
 - **IDirect SoundBuffer**
 - **IDirect Sound3DBuffer**
 - **IDirect Sound3DListener**
 - **IDirect SoundCapture**
 - **IDirect SoundCaptureBuffer**

DirectX - DirectPlay

- Simplifica el acceso de las aplicaciones a los servicios de comunicación
- Otorga independencia para la creación de servidores de juegos
- Comunicaciones
 - Peer-to-Peer
 - Cliente/Servidor

DirectX - Direct3D

- Interfaz gráfica para hardware 3D
 - Permitir gráficos tridimensionales interactivos en aplicaciones de Windows
- 2 Modos:
 - Inmediato
 - API 3D de bajo nivel
 - Independiente del dispositivo
 - Programadores experimentados
 - Retenido (Obsoleta)
 - Desarrollo rápidos
 - Capa superior del inmediato

DirectX - Direct3D

- Conceptos Básicos
 - Sistemas de coordenadas 3-D
 - Left-handed (mano izquierda)
 - Podemos simular el right-handed
 - Transformaciones 3-D
 - Traslación
 - Rotación
 - Escalado
 - Polígonos
 - Normales de cara y vértice
 - Modos de sombreado
 - Interpolaciones de triángulos

DirectX - Direct3D

- Conceptos Básicos
 - Triángulos
 - Reglas de rasterización de triángulos

OpenGL

- Introducido en 1992 por SGI
- Basado en IRIS GL, un API para workstations SGI
- Es un *open standard* que ha sido adoptado ampliamente para todo tipo de aplicaciones gráficas
- Se desarrolla bajo la supervisión del **OpenGL architecture review board**

Objetivos de diseño de OpenGL:

- API gráfico de altas prestaciones (con aceleración por hardware)
- Posee cierta independencia del hardware
- Es un API natural (en C) con posibilidad de extensibilidad

Se convierte en standard porque ...

- No trata de hacer demasiadas cosas:
 - Sólo renderiza la imagen, no gestiona ventanas, etc...
 - No posee animación de alto nivel, modelado, sonido, etc...
- Hace lo suficiente:
 - Efectos de renderizado útiles y altas prestaciones
- Fue promovido por empresas líderes en el sector (SGI, Microsoft, etc)

Ventajas de OpenGL

- **Industry standard**
An independent consortium, the OpenGL Architecture Review Board, guides the OpenGL specification. With broad industry support, OpenGL is the only truly open, vendor-neutral, multiplatform graphics standard.
- **Stable**
OpenGL implementations have been available for more than **seven years** on a wide variety of platforms. Additions to the specification are well controlled, and proposed updates are announced in time for developers to adopt changes. Backward compatibility requirements ensure that existing applications do not become obsolete.
- **Reliable and portable**
All OpenGL applications produce consistent visual display results on any OpenGL API-compliant hardware, regardless of operating system or windowing system.

Ventajas de OpenGL

- **Evolving**
Because of its thorough and forward-looking design, OpenGL allows new hardware innovations to be accessible through the API via the OpenGL extension mechanism. In this way, innovations appear in the API in a timely fashion, letting application developers and hardware vendors incorporate new features into their normal product release cycles.
- **Scalable**
OpenGL API-based applications can run on systems ranging from consumer electronics to PCs, workstations, and supercomputers. As a result, applications can scale to any class of machine that the developer chooses to target.

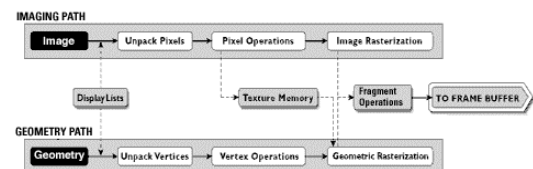
Ventajas de OpenGL

- **Easy to use**
OpenGL is well structured with an intuitive design and logical commands. Efficient OpenGL routines typically result in applications with fewer lines of code than those that make up programs generated using other graphics libraries or packages. In addition, OpenGL drivers encapsulate information about the underlying hardware, freeing the application developer from having to design for specific hardware features.
- **Well-documented**
Numerous books have been published about OpenGL, and a great deal of sample code is readily available, making information about OpenGL inexpensive and easy to obtain.

Renderizado de OpenGL

- Primitivas geométricas:
 - Puntos, líneas y polígonos
- Primitivas de imágenes:
 - Imágenes y bitmaps
- Pipelines separados para imágenes y geometría unidos mediante el mapeador de texturas
- El renderizado depende del estado (luces, colores, materiales, etc)

Arquitectura OpenGL



Capacidades de OpenGL

- **Accumulation buffer** A buffer in which multiple rendered frames can be composited to produce a single blended image. Used for effects such as depth of field, motion blur, and full-scene anti-aliasing.
- **Alpha blending.** Provides a means to create transparent objects.
- **Automatic rescaling of vertex normals** changed by the modeling matrix.
- **BGRA pixel formats and packed pixel formats** to directly support more external file and hardware frame buffer types.
- **Color-index mode.** Color buffers store color indices rather than red, green, blue, and alpha color components.
- **Immediate mode.** Execution of OpenGL commands when they're called, rather than from a display list.
- **Display list.** A named list of OpenGL commands. The contents of a display list may be preprocessed and might therefore execute more efficiently than the same set of OpenGL commands executed in immediate mode.

Capacidades de OpenGL

- **Double buffering.** Used to provide smooth animation of objects. Each successive scene of an object in motion can be constructed in the back or "hidden" buffer and then displayed. This allows only complete images to ever be displayed on the screen.
- **Feedback.** A mode where OpenGL will return the processed geometric information (colors, pixel positions, and so on) to the application as compared to rendering them into the frame buffer.
- **Level of detail control** for mipmap textures to allow loading only a subset of levels.
- **Materials lighting and shading.** The ability to accurately compute the color of any point given the material properties for the surface.
- **Pixel operations.** Storing, transforming, mapping, zooming.
- **Polynomial evaluators.** To support non-uniform rational B-splines (NURBS).
- **Primitives.** A point, line, polygon, bitmap, or image.
- **Raster primitives.** Bitmaps and pixel rectangles.

Capacidades de OpenGL

- **RGBA mode.** Color buffers store red, green, blue, and alpha color components, rather than indices.
- **Selection and picking.** A mode in which OpenGL determines whether certain user-identified graphics primitives are rendered into a region of interest in the frame buffer.
- **Specular Highlights.** Application of specular highlights after texturing for more realistic lighting effects.
- **Stencil planes.** A buffer used to mask individual pixels in the color frame buffer.
- **Texture coordinate edge clamping** to avoid blending border and image texels during texturing.

Capacidades de OpenGL

- **Texture mapping.** The process of applying an image to a graphics primitive. This technique is used to generate realism in images.
- **Three Dimensional Texturing.** Three-dimensional texturing for supporting hardware-accelerated volume rendering.
- **Transformation.** The ability to change the rotation, size, and perspective of an object in 3D coordinate space.
- **Vertex array enhancements** to specify a subrange of the array and draw geometry from that subrange in one operation.
- **Z-buffering.** The Z-buffer is used to keep track of whether one part of an object is closer to the viewer than another.

APIs relacionadas

- GLX, WGL, AGX
 - Conexiones de OpenGL con el entorno de ventanas
- GLU (OpenGL Utility library)
 - Es parte de OpenGL
 - Incluye soporte para cuádricas, NURBS, etc.
- GLUT (OpenGL Utility Toolkit)
 - No forma parte oficialmente de OpenGL
 - Permite la portabilidad de las aplicaciones sobre distintos sistemas de ventanas
- MESA: OpenGL clone

APIs de OpenGL

- Librería de funciones para generar imágenes a partir de modelos 3D, más otras librerías auxiliares
 - gl la librería opengl relacionada directamente con el H/W
 - glu librería de mayor nivel construida sobre opengl
 - glaux librería fuera de uso
 - glut librería para crear interfaces de usuario transportables (Linux, Windows, Unix en general, MacOS)
 - glut.dll → windows\system(32)
 - glut.lib → DevStudio\vc\lib
 - glut.h → DevStudio\vc\include\gl

Arquitectura de APIs de OpenGL



Detalles de programación

- Añadir las librerías opengl32.lib glu32.lib glut32.lib
 - Project : settings ; link : Object/Library Modules
- Ficheros a incluir
 - #include <GL/gl.h>
 - #include <GL/glu.h>
 - If you are using GLUT for managing your window manager
 - #include <GL/glut.h>
 - Note that glut.h includes gl.h, glu.h, and glx.h automatically, so including all three files is redundant.

Estados

- Máquina de estados
 - Color de fondo
 - Intensidad de luz
 - Material de dibujo
 - Luz encendida o apagada
 - etc
- Valor o estado
 - glColor*(), glGetFloatv()
 - glEnable(), glDisable(), glIsEnabled()

Command syntax (functions)

- glVertex3fv(...)
 - **gl** tells that this function belongs to the “gl” s/w package
 - **3** is used to indicate three arguments
 - **f** is used to indicate that the arguments are floating point
 - **v** indicates that the arguments are in vector format
- Number Of Arguments: 2, 3, or 4
 - Bi-dimensional version of the command
 - 3D or rgb
 - Homogeneous coordinates or rgb+alpha
- Formats
 - absence of v indicates scalar format
 - v indicates vector format

Tipos variables y constantes

- Las equivalencias más habituales aparecen en la tabla
 - Se recomienda definir los argumentos que se pasan a las funciones de OpenGL usando sus tipos predefinidos
 - Para GLint unos sistemas pueden usar short, otros long
 - Para GLfloat unos sistemas float otros double

	Data type	Typical Corresponding C-Language Type	OpenGL Type Definition
b	8-bit integer	C-Language Type	GLbyte
s	16-bit integer	OpenGL Type	GLshort
i	32-bit integer	int or long	GLint GLsizei
f	32-bit floating-point	float	GLfloat GLclampf
d	64-bit floating-point	double	GLdouble GLclampd
ub	8-bit unsigned integer	unsigned char	GLubyte GLboolean
us	16-bit unsigned integer	unsigned short	GLushort
ui	32-bit unsigned integer	unsigned int or unsigned long	GLuint GLenum GLbitfield

Constantes

- Ejemplo : GL_COLOR_BUFFER_BIT
- Defined constants
 - Begin with GL_
 - Use all capital letters
 - Use underscores to separate words
- Con frecuencia se realizan operaciones “or”

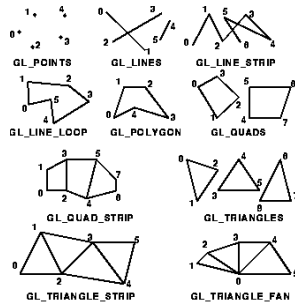
... (gl)

- pure "output", but lacks connection with display
 - glClear (command)
 - glClearColor (state)
 - glBegin glEnd (geometry)
 - glVertex* (attribute, state)
 - glColor* (command stack & processing)
 - glFlush, glFinish (debugging)
 - (mapping from modelling coord. to display coordinates)
 - (aspect ratio)
 - (need for a transformation management system)

... (glut)

- Window system independent management
 - "window" management
 - glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
 - glutInitWindowSize(WIDTH, HEIGHT);
 - glutInitWindowPosition (550, 350);
 - glutCreateWindow("Basic Draw");
 - "input": event handling
 - similar to : The X Window system, MOTIF, MFF, ... (event)
 - void Display(void); (callback function)
 - glutDisplayFunc(Display); (binding event & callback)
 - void Display (void)
 - glutMainLoop(); (dispatching events)

Primitivas



...

- Otras primitivas disponibles
 - Objetos (uso auxiliar, no para crear modelos)
 - Vertex arrays
 - Display lists
 - Evaluators, NURBS, etc
- Texto
 - No existe primitiva
 - Usar texturas, problema aliasing

Atributos

- glPointSize(GLfloat)
- glLineWidth(GLfloat)
- glLineStipple(GLint factor, GLushort pattern)
 - glEnable(GL_LINE_STIPPLE)
- glPolygonMode(face, mode)
 - GL_FRONT GL_BACK GL_FRONT_AND_BACK
 - GL_POINT GL_LINE GL_FILL
- glPolygonStipple (enable) "transparency"
- glEdgeFlag* mode Line, splitted concave polygons
- glColor*

PATTERN	FACTOR
0x00FF	1
0x00FF	2
0x0C0F	1
0x0C0F	3
0xAAAA	1
0xAAAA	2
0xAAAA	3
0xAAAA	4
- glMaterial*

PATTERN	FACTOR
0x00FF	1
0x00FF	2
0x0C0F	1
0x0C0F	3
0xAAAA	1
0xAAAA	2
0xAAAA	3
0xAAAA	4

Variables de estado

- Valor de defecto
- Valor actual
- Gestión en una aplicación o en un módulo reutilizable
 - Valor de defecto en una aplicación dada
 - Política de cambios de estado en un módulo de software
- glPushAttrib(ored mask) / glPopAttrib()
 - Permiten salvar grupos de atributos

Grupos de atributos

GL_ACCUM_BUFFER_BIT	accum-buffer
GL_ALL_ATTRIB_BITS	--
GL_COLOR_BUFFER_BIT	color-buffer
GL_CURRENT_BIT	current
GL_DEPTH_BUFFER_BIT	depth-buffer
GL_ENABLE_BIT	enable
GL_EVAL_BIT	eval
GL_FOG_BIT	fog
GL_HINT_BIT	hint
GL_LIGHTING_BIT	lighting
GL_LINE_BIT	line
GL_LIST_BIT	list
GL_PIXEL_MODE_BIT	pixel
GL_POINT_BIT	point
GL_POLYGON_BIT	polygon
GL_POLYGON_STIPPLE_BIT	polygon-stipple
GL_SCISSOR_BIT	scissor
GL_STENCIL_BUFFER_BIT	stencil-buffer
GL_TEXTURE_BIT	texture
GL_TRANSFORM_BIT	transform
GL_VIEWPORT_BIT	viewport

Agrupamiento

- Arrays
 - Evitar pérdidas de tiempo en llamadas a funciones
- Display Lists
 - Ídem
 - La información puede estar preprocesada
 - Almacenar en el procesador gráfico para aliviar el bus
 - Workstations
 - PCs

Comandos entre : glBegin-glEnd

Command	Purpose of Command	Reference
glVertex*()	set vertex coordinates	Chapter 2
glColor*()	set current color	Chapter 5
glIndex*()	set current color index	Chapter 5
glNormal*()	set normal vector coordinates	Chapter 2
glEvalCoord*()	generate coordinates	Chapter 11
glCallList(), glCallLists()	execute display list(s)	Chapter 4
glTexCoord*()	set texture coordinates	Chapter 9
glEdgeFlag*()	control drawing of edges	Chapter 2
glMaterial*()	set material properties	Chapter 6

Ejemplo : Window to viewport

- gl
 - glMatrixMode
 - glLoadIdentity
 - viewport
 - glOrtho
 - glGetIntegerv
- glut
 - glutReshapeFunc(myNewSize)
 - void myNewSize(int w, int h)

Links Interesantes

- www.opengl.org Official site of OpenGL
- nehe.gamedev.net NeHe, various OpenGL tutorials
- romka.demonews.com Romka, various OpenGL tutorials
- Nexe.gamedev.net same as Nehe for DirectX