Reporte de Búsqueda de Bibliotecas de funciones Gráficas, de Alto Rendimiento y de Dominio Público en INTERNET

Javier Gutiérrez[1], **y Violeta Rangel**[2]
Centro Nacional de Cálculo Científico
Universidad de Los Andes (CeCalCULA),
Corporación Parque Tecnológico de Mérida, Mérida 5101, Venezuela

Rubén Medina [3]
Grupo de Bioingeniería,
Departamento de Electrónica, Facultad de Ingeniería,
Universidad de Los Andes, Mérida 5101, Venezuela y
Centro Nacional de Cálculo Científico
Universidad de Los Andes (CeCalCULA),
Corporación Parque Tecnológico de Mérida, Mérida 5101, Venezuela

Luis A. Núñez [4]
Centro de Astrofísica Teórica,
Departamento de Física, Facultad de Ciencias,
Universidad de Los Andes, Mérida 5101, Venezuela, y
Centro Nacional de Cálculo Científico
Universidad de Los Andes (CeCalCULA),
Corporación Parque Tecnológico de Mérida, Mérida 5101, Venezuela

Versión α 2.0, Junio 2000

[1]gutierre@cecalc.ula.ve
[2]rangelv@cecalc.ula.ve
[3]rmedina@ing.ula.ve
[4]nunez@ciens.ula.ve

Resumen

En este reporte se hace una breve introducción a los conceptos de computación gráfica más relacionados con la visualización científica y se hace una descripción de las principales bibliotecas de desarrollo para ambientes gráficos. Se incluye también hace una descripción de seis bibliotecas de funciones gráficas de alto nivel especializadas en la visualización científica, haciendo incapié en la generación de vizualizaciones en 3 dimensiones con *render* de iso-superficies y representaciones volumétricas.

Índice

1. Introducción

Debido a la forma de crecimiento de la INTERNET, se ha ido formando en ella un gigantesco repositorio de herramientas para la programación. Sin embargo, la mayoría de dichas herramientas se presentan sin la documentación necesaria para el máximo aprovechamiento por parte de usuarios ajenos al proyecto que las generó. Es por eso que se pretende en hacer una colección de funciones, de dominio público, para resolver problemas de visualización de datos multidimensionales.

Esta colección será evaluada y documentada, *a posteriori*, para permitir a los usuarios llegar rápidamente a la rutina que necesitan para resolver su problema y saber cuales son las capacidades y limitaciones de dichas rutinas. La idea detrás de esto es que se pueda reducir el tiempo que transcurre entre la concepción de una idea, hasta la implementación de ésta en una plataforma, es decir, que se pueda pasar con mínimo esfuerzo de los prototipos desarrollados en aplicaciones tipo **MatLab** a programas escritos en **C** y **FORTRAN** con su interfaz gráfica.

Este documento está organizado de la siguiente manera. A continuación se hace una descripción de algunos tópicos de la computación gráfica. En la sección 2 se hace una breve descripción de las principales bibliotecas gráficas de interacción directa con el *hardware*. En la seccion 3 se describen algunas bibliotecas de alto nivel que permiten realizar gráficos de visualización científica en 2D y 3D.

Dentro de la computación gráfica existe una enorme cantidad de términos que suelen ser mencionados con poca precisión. A continuación se definen algunos de estos términos que serán utilizados como referencia para describir las capacidades de las bibliotecas de funciones de alto nivel.

- **Profundidad de Imagen**: Este es un término que indica el número de colores disponible para pintar cada pixel de la imagen. Esta profundidad está medida en bits por pixel (bpp): por ejemplo se tiene que una imagen de 8bpp tiene 256 colores, mientras que una imagen de 24bpp tiene aproxiamdamente 16 millones de colores disponibles para cada pixel.

- ***Volumen Rendering***: Bajo este nombre se agrupa a una serie de técnicas utilizadas para hacer representaciones gráficas tridimensionales de datos volumétricos. Entre las técnicas más utilizadas se encuentran *Volumen Slicing*, Sombreado por Profundidad (*Depth-only Shading*), trazado de rayos *Raytracing*, *Ray Casting*.

- **Isosuperficies**: La representación de iso-superficies es la generalización a tres dimensiones de las isofotas o líneas de contorno utilizadas en cartografía y otras representaciones en dos dimensiones. Esta es una de las técnicas más utilizadas para la visualización de campos escalares en tres dimensiones. Entre las técnicas más utilizadas para la generación y representación de iso-superficies está el algoritmo de *Marching Cubes*.

- **Stream Lines**: Las líneas de flujo son una de las técnicas de representación de campos vectoriales en tres dimensiones, utilizada con preferencia para campos de velocidades, y que está basada en la construcción de "tubos" que pasan a través de las zonas donde los vectores tienen el mismo módulo visualizando fácilmente la formación de turbulencias y remolinos.

- **Z-bufer**: El Z-buffer es el nombre que se le da a una matriz bidimencional que indica la distancia que hay desde el plano de proyección de la imagen hasta el *voxel* (elemento de volumen) más cercano. Esto establece un mapa de profundidad de los objetos en la imagen, esta información es utilizada en algoritmos como el Sombreado por Profundidad y el *Ray Casting*. Algunas tarjetas de aceleración gráfica permiten hacer el cálculo y actualización del Z-buffer via *hardware*.

2. Bibliotecas de interfaz con el *hardware*

Estas bibliotecas están constituidas, generalmente, por rutinas para generar primitivas geométricas muy elementales, para permitir al usuario el desarrollo de rutinas que permitan la aplicación de algoritmos de visualización, render y fotorealismo. Estas bibliotecas pueden estar o no ligadas directamente con un *hardware* en particular, pero en general están desarrolladas sobre un estándar dejando al fabricante del *hardware* la implementación de la interfaz con la biblioteca.

2.1. OpenGL

OpenGL es un *software* de interfaz con el *hardware* gráfico. Esta interfaz consiste de 150 comandos o rutinas que permiten especificar las características de los objetos en el espacio.

Esta biblioteca de rutinas esta diseñada para el desarrollo de aplicaciones gráficas interactivas y es independiente del *hardware*, por lo tanto, OpenGL no hace manejo de ventanas ni de interfaces de usuario. Además OpenGL no es una biblioteca de alto nivel, ya que no posee rutinas que permitan la descripción de objetos complejos, de hecho las *primitivas* del OpenGL son: puntos, líneas y polígonos.

2.1.1. Capacidades de Graficación y Render

OpenGL permite la descripción matemática de objetos a partir de sus primitivas. Permite la organización de estos objetos en el espacio al tiempo que permite modificar el punto de vista y las propiedades de la cámara.

Permite calcular el color de los pixeles por asignación directa de color, por cálculo de iluminación, por mapeo de texturas o por una combinación de los tres.

Permite convertir la descripción matemática de objetos en formas y colores. Permite también la simulación de "efectos atmosféricos" como cubo de profundidad, transparencia, neblina y render volumétrico.

2.1.2. Capacidad de desarrollo de interfaz

OpenGL no permite el desarrollo de interfaces gráficas por si solo. Sin embargo existen algunas bibliotecas de alto nivel desarrolladas sobre OpenGL que facilitan esta labor.

2.1.3. Requerimientos de *hardware* y *software*

El OpenGL no requiere en particular ningún *hardware*, pero existe en el mercado una gran cantidad de tarjetas aceleradoras gráficas que incluyen la rutinas del OpenGL en el *hardware*, lo que incrementa notablemente la eficiencia en gráficos en aplicaciones de tiempo real.

El OpenGL requiere de compiladores C, C++ y de FORTRAN que puedan llamar rutinas compiladas en C. Normalmente se requiere una distribución de OpenGL que incluya el sofware que lo hace compatible con el ambiente gráfico que se está utilizando, por ejemplo GLX para correr en ambiente X o WinGL para correr en máquinas Win9x o win NT.

2.1.4. Portatilidad

La biblioteca OpenGL está desarrollada en ANSI C para garantizar la compilación en diferentes plataformas. Además existen una serie de versiones precompiladas para Windows, Macintosh y ambiente X.

2.1.5. Documentación

La documentación e información sobre las versiones comerciales y de dominio público de la biblioteca **OpenGL** se encuentra en el sitio web *http://www.opengl.org*.

2.1.6. Mesa 3-D Graphics Libray

Mesa[1] es una biblioteca gráfica de dominio público con una interfaz basada en **OpenGL** con autorización de **Silicon Graphics Inc**. Casi todos los programas escritos en base a la biblioteca **OpenGL** funcionan sin cambios con la biblioteca **Mesa**, sin embargo el autor[2] dice que **Mesa** no es un reemplazo completo del **OpenGL**, por este motivo existe un documento en el web[3] donde se especifican las pruebas de compatibilidad entre estas dos bibliotecas gráficas.

2.2. CAVElib

Es una biblioteca de funciones para controlar el *hardware* de Realidad virtual de los sistemas de proyeccion **CAVE**[4], **ImmersaDesk**[5] e **InfinityWall**[6].

2.2.1. Capacidades de Graficación y Render

Entre las capacidades de control de *hardware* se encuentran:

- Control de lentes estéreo.
- Control de proyectores estéreo.
- Interacción con la proyección controlada por la posición y los movimientos del usuario.
- Navegación
- Está diseñada para tener una interfaz natural con rutinas de graficación provistas por el usuario.

2.2.2. Capacidad de desarrollo de interfaz

CAVElib posee todas las rutinas y las macros necesarias para controlar la interacción del usuario con la proyección, así como para la navegación a través de la escena.

2.2.3. Aplicaciones

Los sistemas **CAVE** e **ImmersaDesk** son utilizados ampliamente en los grupos de visualización de los centros más prestigiosos, en el sitio web *http://www.evl.uic.edu/pape/CAVE/sites.html* se presenta una lista completa de las instalaciones actuales de estos sistemas y de sus aplicaciones.

2.2.4. Documentación

La documentación sobre la biblioteca **CAVElib** se encuentra en la dirección web: http://www.evl.uic.edu/pape/

[1] http://www.mesa3d.org/
[2] http://www.mesa3d.org/devel.html#Author
[3] http://www.mesa3d.org/lxr/source/docs/CONFORM
[4] http://www.evl.uic.edu/pape/CAVE/
[5] http://www.evl.uic.edu/pape/CAVE/idesk
[6] http://www.evl.uic.edu/pape/CAVE/images/wall.jpg

3. Bibliotecas de Alto Nivel

En esta sección se agrupan las bibliotecas que proveen rutinas de alto nivel para aplicar algoritmos de visualización sobre estructuras de datos estándar. Estas bibliotecas suelen estar construidas sobre las bibliotecas de interacción con el *hardware*.

3.1. VTK - Visualization ToolKit

El **Visualization ToolKit (VTK)** es un software de dominio público, distribuido bajo el modelo *Open Source*, para computación gráfica en 3D, procesamiento de imágenes y visualización. El diseño de esta biblioteca está fuertemente basado en objetos.

3.1.1. Capacidades de Graficación y Render

VTK es un sistema real de visualización ya que incluye capacidades para desplegar primitivas geométricas al tiempo que incluye algoritmos para la visualización de campos escalares, campos vectoriales, visualización de tensores, mapeo de texturas, así como métodos volumétricos para representación de iso-superficies y volúmenes.

VTK incluye además técnicas avanzadas de modelado como lo son la reducción de poligonos, suavizado de mallas, triangulación de Delauny, etc. Cuenta también con una amplia gama de rutinas para manipulación de imágenes, lo que permite incluir imágenes 2D en las escenas 3D en formas de mapas, texturas o imagenes de fondo.

Existe una lista completa de las capacidades de esta biblioteca en la pagina hogar de **VTK**[7]

3.1.2. Capacidad de desarrollo de interfaz

Posee un conjunto de capas de enlace con los lenguajes **Tcl/Tk** y **Python/Tk** y con el **Java**, lo cual permite desarrollar la interfaz del usuario con alguno de estos lenguajes.

3.1.3. Aplicaciones

La biblioteca **VTK** está siendo utilizada activamente en la comunidad científica y se consigue referencia a ella en aplicaciones diversas:

- MapInfo: *http://www.mapinfo.com*
- Visualization of Diffpack Simulations: *http://www.ifi.uio.no/ xingca/vtk_diffpack/*
- Simulating Acoustic Fields: *http://cnmat.CNMAT.Berkeley.EDU/AcousticVisualization/*
- National Library of Medicine Visible Human *http://www.crd.ge.com/esl/cgsp/projects/makevw/*
- NCSA Cave Visualization *http://brighton.ncsa.uiuc.edu/ prajlich/vtkActorToPF/*
- Geocap Geoscience Environment *http://www.geocap.no*

[7] http://www.kitware.com/vtkhtml/vtkdata/WhatIsVTK.html

3.1.4. Requerimientos de *hardware* y *software*

El **VTK** esta desarrollada completamente en C++, y tiene desarrollada las interfaces para utilizar las rutinas desde **Tcl**, **Python**, y **Java**.

La instalación de **VTK** requiere soporte para **OpenGL**, lo cual puede ser con una versión comercial de **OpenGL** o con algunas de las distribuciones de dominio público incluida la distribución **MESA** (versión 3.0-2 o superior).

Igualmente si se desea utilizar la interfaz con **Tcl/Tk** se debe instalar la versión 8.0.2-16 o superior.

Por otro lado la biblioteca de funciones de **VTK** es bastante grande por lo que se recomienda el uso de *hardware* de aceleración gráfica con soporte para **OpenGL** y Z-buffer, así como el uso de una computadora multiprocesador con suficiente RAM para manejar los datos.

3.1.5. Portatilidad

Cualquier plataforma Unix, PC, o Mac con compilador de C++ y soporte para OpenGL.

3.1.6. Documentación

La documentación de **VTK** incluye un libro publicado por Prentice-Hall (The Visualization Toolkit, An Object-Oriented Approach To 3D Graphics, 2nd edition ISBN 0-13-954694-4). y está disponible en el Web[8].

3.2. DrawP3D

DrawP3D es una biblioteca de subrutinas para realizar visualización científica. Fue desarrollada inicialmente sobre el formato de objetos P3D, sin embargo, ahora puede exportar a los formatos VRML y Open Inventor (SGI), además del original P3D.

Tambien provee la posibilidad de salidas interactivas tanto a la intefaz **X** como al ambiente **Silicon Graphics GL**.

3.2.1. Capacidades de Graficación y Render

Este paquete de rutinas puede producir iso-superficies, superficies Z, así como las primitivas básicas de polígonos, esferas y cilindros.

DrawP3D puede manejar datos en mallas regulares, en mallas irregulares o esparcidos aleatoriamente.

DrawP3D no soporta en la actualidad el mapeo de texturas, pero tiene amplias capacidadess de aplicación de mapas de colores. Este tipo de capacidades pueden ser utilizadas para colorear una iso-superficie en base a un campo adicional, permitiendo la visualización de una segunda variable.

Además de las capacidades gráficas **DrawP3D** se puede ejecutar en forma distribuida utilizando la biblioteca de pase de mensajes **PVM** y puede interactuar con *hardware* de realidad virtual tipo **CAVE**.

3.2.2. Capacidad de desarrollo de interfaz

DrawP3D no provee rutinas para el desarrollo de interfaz con el usuario, sin embargo puede ser utilizada desde los lenguajes de *script* tipo **Tcl/Tk**.

[8] http://www.kitware.com/vtk.html

3.2.3. Requerimientos de *hardware* y *software*

La biblioteca de pase de mensajes **PVM** es requerida para correr en forma distribuida. Igualmente se requiere de los paquetes **SGI Open Inventor** y **SGI Performer** si se desea utilizar la interfaz interactiva en ambiente **SGI GL**. La plataforma **SGI Performer** también es requerida para interactuar con los sistemas tipo **CAVE**

3.2.4. Portatilidad

La biblioteca **DrawP3D** debe ser compilada sobre una plataforma **UNIX**. Hasta ahora ha sido probada en **SGI**, **Cray**, **Sun**, **DEC**, y **AIX**.

Para **Linux**, se debe construir un archivo "Makefile" para compilar el paquete. El "visualization server" que corre remotamente en modo distribuido no funciona sobre **Linux** ya que para esto se requiere **Open Inventor** o **SGI Performer** pero el paquete aún puede ser muy útil sin esta funcionalidad.

3.2.5. Documentación

La documentación puede ser obtenida en el web a través de los protocolos http o ftp:

- ftp://ftp.psc.edu/pub/drawp3d/
- http://www.psc.edu/Packages/DrawP3D_Home/

3.3. MAM-VRS

MAM (*Modeling and Animation Machine*) es un conjunto de utilidades para gráficos 3D animados e interactivos y **VRS** (*Virtual Rendering System*) es un sistema de Rendering que encapsula bibliotecas tipo **OpenGL** en una interfaz uniforme orientada a objetos.

3.3.1. Capacidades de Graficación y Render

La bibilioteca MAM/VRS tiene rutinas que facilitan las técnicas de *Volumen Rendering*, *Glyph*, etc. en 3D.

Posee también rutinas para mapeo de texturas, manipulación de imágenes, e interactividad.

MAM/VRS puede producor salidas a varios sistemas de render, incluidos **OpenGL**, **POVRay**[9], entre otros.

3.3.2. Capacidad de desarrollo de interfaz

Puede ser utilizado desde el **Tcl/Tk** para construir la interfaz del usuario.

3.3.3. Aplicaciones

3.3.4. Requerimientos de *hardware* y *software*

MAM/VRS no posee rutinas para el desarrollo de una interfaz de usuario, pero puede ser utilizados desde la mayoria de las bibliotecas desarrolladas con este fin, como lo son, **Motif**, **LessTif**, **Tcl/Tk**, etc.

[9] http://www.povray.org

3.3.5. Portatilidad

MAM/VRS ha sido compilado y probado en:

- SUN Solaris (2.7)
- SGI (IRIX >= 6.3)
- Linux
- Windows 95/98/NT (DeveloperStudio 5.0/6.0, MFC, Tcl/Tk >=8.0 for Windows)

3.3.6. Documentación

3.4. OpenRM

OpenRM es un conjunto de subrutinas, de alto nivel, para la construcción de escenas que utilizan al **OpenGL** como plataforma gráfica, estas subrutinas están diseñadas para aprovechar el *hardware* de aceleración gráfica. Entre las características más resaltantes de **OpenRM** está la posibilidad de interactuar con rutinas especificas de *hardware* de visualización como la **CAVElib**.

3.4.1. Capacidades de Graficación y Render

Entre las capacidades de **OpenRM** se pueden citar:

- 3D Opaco y transparente
- Los objetos pueden heredar todos los atributos de otros objetos en la escena
- Soporta varios métodos de visualización en estéreo
- Tiene soporte para render volumétrico directo.
- Puede incluir varias vistas en una sola ventana y/o sincronizar los relojes de varias ventanas para control de animación.
- En compatible con los ambientes X11 y Win32.
- Permite control directo sobre los *buffer* de imagen y de *background*
- Permite la utilización de planos de corte.

3.4.2. Capacidad de desarrollo de interfaz

El **OpenRM** no incluye un núcleo para el desarrollo de una interfáz de usuario, sin embargo incluye rutinas que permiten al usuario seleccionar, con el ratón u otro dispositivo de entrada, uno o varios objetos en la escena que se está visualizando.

3.4.3. Requerimientos de *hardware* y *software*

La instalación de **OpenRM** requiere la instalación de la biblioteca **OpenGL** y se recomienda la utilización de *hardware* de aceleración gráfica con capacidad para desplegar imágenes en 32bpp y con soporte para *Z-buffer*

3.4.4. Portatilidad

OpenRM está diseñada para que sea compatible con los ambientes gráficos X11 y Win32, lo que le permite ser utilizado en MS-Windows y en las plataformas Unix.

OpenRM puede ser utilizada desde C y Fortran.

3.4.5. Documentación

La documentación de esta biblioteca está disponible en el web[10]

3.5. VolPack

VolPack es una biblioteca de rutinas portátiles diseñadas para render volumétrico y está basada en una familia de nuevos algoritmos que permiten el render rápido de volumenes. (see Philippe Lacroute and Marc Levoy , Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation, Proceedings of SIGGRAPH94).

La bilioteca se distribuye con una aplicación de ejemplo que utiliza la interfaz de usuario desarrollada en el lenguaje de script **Tcl/Tk** y el motor gráfico desarrollado completamente en **VolPack**

3.5.1. Capacidades de Graficación y Render

Entre las características de **VolPack** están:

- Visualiza datos sobre una malla regular tridimensional.
- Permite el uso de funciones definidas por el usuario para la opacidad y el color
- Provee un modelo de sombreado con luces direcionales, materiales y reflexiones.
- Produce render en escala de grices o en colores (24bpp) con o sin canal alfa.
- Soporta transformaciones afines arbitrarias

3.5.2. Capacidad de desarrollo de interfaz

La biblioteca **VolPack** no posee rutinas de creación de interfaz de usuario, esta puede ser desarrollada con alguna de las bibliotecas para este fin como **Tcl/Tk** o **Python/Tk**.

3.5.3. Aplicaciones

3.5.4. Requerimientos de *hardware* y *software*

La biblioteca **VolPack** está diseñada para ser utilizada en los lenguajes **C** y **C++**, pero puede ser utilizada tambien desde otros lenguajes.

Esta biblioteca no hace uso del *hardware* de aceleración gráfica, lo que la hace portatil entre la mayoría de la plataformas UNIX.

3.5.5. Documentación

La documentación de la biblioteca está disponible en el web, *http://www-graphics.stanford.edu/software/volpack*

[10] http://openrm.sourceforge.net/docs/index.shtml

3.6. Math3d

Math3d es una biblioteca diseñada para simplificar la notación a la hora de programar una aplicación gráfica.

3.6.1. Capacidades de Graficación y Render

La biblioteca **Math3d** provee las definiciones de las clases:

- M2d (2-dimensional vector)
- M3d (3-dimensional vector)
- M4d (4-dimensional vector)
- MQuat (Quaternion)
- MRot (Rotation in Axis/Angle representation)
- M4x4 (4x4 homogenous matrix)
- M3Frame (Frenet 3-frame)
- MLookAt (Frenet 3-frame + distance (==target))
- M2dRect (2-dimensional Rectangle)
- M3dBox (Axis aligned bounding box)

3.6.2. Aplicaciones

Esta biblioteca está siendo utilizada en los siguientes proyectos:

- 3D Studio File Library (lib3ds) :*http://lib3ds.sourceforge.net/*
- Digital Actor: *http://digitalactor.sourceforge.net/*
- Free Animation and Modeling Project (FAMP): *http://famp.sourceforge.net/*
- Rendering Animation and Modeling Architecture (RAMA): *http://rama.sourceforge.net/*

3.6.3. Portatilidad

Esta biblioteca está siendo desarrollada en los ambientes Linux y Win32.

3.6.4. Documentación

La documentación de esta biblioteca está incluida en la distribución que se obtiene de la red[11].

[11] http://math3d.sourceforge.net/