Creación Interactiva de Grafos de Escena para Aplicaciones Gráficas 3D

Scene Graph Interactive Creation for 3D Graphics Applications

Juan Camilo Ibarra López, Ing., Fernando De la Rosa, PhD.
Grupo de Investigación IMAGINE, Departamento de Ingeniería de Sistemas y Computación
Universidad de los Andes, Bogotá, Colombia
{ju-ibarr, fde}@uniandes.edu.co

Recibido para revisión 16 de Enero de 2008, aceptado 14 de Febrero de 2008, versión final 27 de Febrero de 2008

Resumen—**Este trabajo presenta una herramienta computacional que permite la creación interactiva de modelos geométricos compuestos 3D basados en el concepto de Grafo de Escena. En un espacio de aprendizaje, el propósito es que el estudiante utilice la herramienta para construir incrementalmente un grafo de escena de un objeto geométrico compuesto 3D de interés y vaya verificando su resultado con una retroalimentación gráfica. Los grafos de escena resultantes se podrán reutilizar en aplicaciones que integren APIs gráficos 3D.**

Palabras Clave—**Aplicaciones, Aprendizaje Activo, Computación Gráfica, Grafo de Escena.**

Abstract—**This work presents a computational tool that leads the interactive creation of 3D geometrical models based on the Scene Graph concept. Into an active learning environment, the purpose is that the student uses the tool for an incremental construction of a 3D object Scene Graph and verifies the results incrementally with a graphical feedback. The obtained scene graphs could be reused in applications that use 3D graphic APIs.**

Keywords—**Applications, Active Learning, Computer Graphics, Scene Graph.**

I. INTRODUCTION

La computación gráfica se basa en el procesamiento computacional de escenas 2D y 3D compuestas por modelos geométricos que poseen detalles en sus superficies y posibles comportamientos en el tiempo. El procesamiento de los modelos geométricos, utilizando un motor gráfico, genera su imagen al interior de la escena que se muestra en una pantalla de computador con características que ofrecen un buen nivel de realismo a los usuarios. Para la definición y representación de modelos geométricos compuestos de otros mas simples (por ejemplo, un avatar compuesto por tronco, cabeza, brazos y piernas) el diseñador de la escena recurre al concepto de grafo de escena [6]. Un grafo de escena representa un diseño de un objeto geométrico que incluye objetos de diferentes niveles en una jerarquía, las relaciones entre estos objetos y facilita su manipulación ya sea por agrupaciones o individualmente. Por la variedad de motores gráficos y de aplicaciones computacionales que trabajan con modelos geométricos, un grafo de escena que defina un objeto geométrico compuesto puede ser traducido a diferentes lenguajes de descripción geométrica (por ejemplo, OpenGL, JOGL, DirectX o VRML).

La importancia del concepto de grafo de escena, su representación y su descripción bajo diferentes lenguajes geométricos es la motivación para crear una aplicación interactiva en computador con el propósito de facilitar al usuario, estudiante en primer lugar o diseñador en segundo lugar, la construcción de grafos de escena que representen objetos geométricos compuestos y su posterior traducción a un lenguaje de descripción geométrica (figura 1).

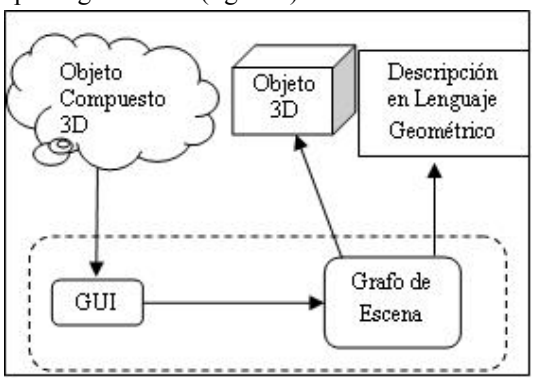

Figura 1. Idea principal del trabajo

La organización de este artículo es la siguiente: la sección 2 presenta la definición de grafo de escena y algunos trabajos de investigación relacionados. En la sección 3 se describe nuestra propuesta de representación de grafos de escena y las diferentes vistas que dispone el usuario para su comprensión. En la sección 4 se ilustran algunos modelos geométricos construidos usando la representación de grafos de escena. Finalmente en la sección 5 se dan las conclusiones del trabajo y mencionan propuestas de trabajos futuros.

II. GRAFOS DE ESCENA: DEFINICIÓN Y TRABAJOS RELACIONADOS

Un grafo de escena es una estructura de datos utilizada en la construcción de representaciones geométricas de objetos complejos la cual ordena lógicamente una serie de nodos que contienen información acerca de un objeto geométrico. En su forma más común el grafo de escena tiene una estructura de árbol n-ario donde hay un único nodo padre y cada nodo puede tener relaciones (arcos) con un conjunto de nodos hijos. Cada nodo en el grafo de escena define un objeto geométrico compuesto por la definición de sus nodos hijos. La aplicación de una transformación geométrica sobre un nodo afecta igualmente a sus hijos. Las transformaciones geométricas clásicas que se requieren en la composición de un objeto geométrico son la traslación, la rotación y el escalamiento (Figura 2).

Figura 2. Grafo de Escena para un objeto geométrico Avatar.

La comprensión de la lógica detrás de la construcción de un grafo de escena permite tener un control adecuado sobre la estructura del objeto (escena) representado a partir de una estructura jerárquica de objetos mas simples teniendo la posibilidad de controlar localmente cambios relativos a un nodo específico (sub-árbol) sin afectar el resto de nodos no contenidos en su descendencia.

Además de los nodos de representación y de transformación también se definen nodos que agrupan otros nodos que comparten una característica en común, normalmente de pertenencia. Estos nodos definen grupos y en algunos casos *layers*, los cuales generan un nuevo nivel de control sobre la escena.

El concepto de grafo de escena es motivo de estudio en cursos de computación gráfica por su utilidad e importancia [4][2]. También existen desarrollos computacionales para el entrenamiento de estudiantes universitarios en el campo de la Computación Gráfica, en especial el de los grafos de escena en donde se le enseña al estudiante la lógica detrás de la construcción de un grafo de este tipo y su funcionalidad [5].

Actualmente, las aplicaciones más comunes de construcción de modelos geométricos utilizan algún tipo de grafo de escena para el control de los cambios de los objetos tanto en 2D como en 3D. Entre estas aplicaciones se destacan como herramientas 3D AutoCad, Solid Edge, 3DStudio Max y Blender; y como herramientas 2D Adobe Illustrator y Corel Draw. Adicionalmente los motores (APIs) gráficos para el desarrollo de nuevos programas permiten definir explícitamente un grafo de escena (por ejemplo, Java3D) o se usa de forma implícita en la descripción de los objetos geométricos (por ejemplo, VRML, OpenGL, DirectX).

También existen librerías especializadas en la construcción de grafos de escena las cuales funcionan sobre algún motor gráfico para representar escenas geométricas. Un ejemplo es OpenSG [10] el cual es un sistema de grafos de escena portable para la construcción de aplicaciones gráficas en tiempo real desarrollado bajo licencia GPL. Otro ejemplo es OpenSceneGraph [11] el cual es un *toolkit* para el desarrollo de aplicaciones 3D de alto desempeño desarrollado en C++ y OpenGL [9].

III. PROPUESTA DE MANEJO DE GRAFOS DE ESCENA

El problema a resolver consiste es permitir a un usuario la construcción de objetos geométricos 3D como una composición de objetos geométricos más simples utilizando como representación un grafo de escena. Los objetos construidos serán traducidos a lenguajes de descripción geométrica para ser integrados en programas de computador. Para cada lenguaje geométrico se requiere conocer en detalle el API de programación específico en el que se hace la traducción.

Un primer paso para la construcción de un objeto geométrico de interés (un mundo) es entonces el diseño del grafo de escena, en donde cada nodo representa un objeto geométrico más simple y los arcos representan las transformaciones entre los nodos. Después de este diseño, el siguiente paso es el de traducir el grafo construido al API del lenguaje de descripción geométrica en el que se quiere visualizar el objeto geométrico de interés.

Para la etapa de diseño del grafo de escena se propone una herramienta computacional que ofrece una GUI de fácil manejo y comprensión. El usuario diseña entonces una jerarquía que contiene la información de un mundo (objeto geométrico) que se desea armar. En la base, se encuentra el directorio de mundos el cual es la raíz de la estructura. Debajo de ella se encuentran 7 tipos de nodos:

- *Directorio*: nodo raíz de la estructura que contiene un

conjunto de mundos (objetos de interés) del usuario.

- *Mundo*: nodo que contiene la información de un grafo de escena que representa un objeto geométrico complejo de interés para el usuario. Un mundo puede hacer parte de otro mundo más complejo a partir de un nodo referencia. Un mundo tiene un conjunto de materiales a disposición de sus elementos geométricos, una secuencia de transformaciones que afectan a todos sus elementos y el grafo de elementos que componen el mundo.

- *Elemento*: nodo de representación de un objeto geométrico que tiene un material asociado e información sobre sus características geométricas. Cada elemento tiene una secuencia de transformaciones que se aplican al elemento y un conjunto de elementos hijo (dependencias) que se encuentran debajo de él en la jerarquía. Por facilidad para el usuario se ofrecen un conjunto de elementos predefinidos:
  - *Caja*: elemento que define un paralelepípedo a partir de su altura, ancho y profundidad y se encuentra centrado en el origen (figura 3).
  - *Esfera*: elemento que define una esfera a partir de su radio la cual se encuentra centrada en el origen (figura 4).

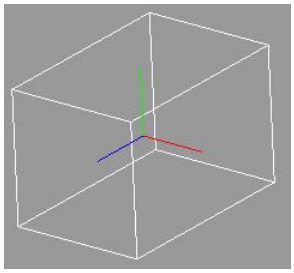
Figura 3. Caja

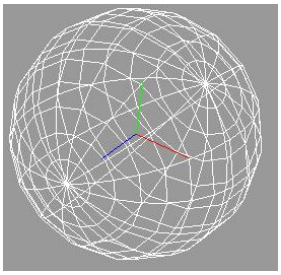
Figura 4. Esfera

  - *Cilindro*: elemento que define un cilindro cónico sin tapas a partir de su radio en la tapa inferior (centrada en el origen), su radio en la tapa superior y su altura definida sobre el eje Z positivo (figura 5).

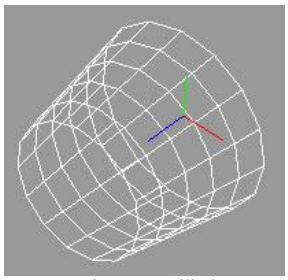
Figura 5. Cilindro

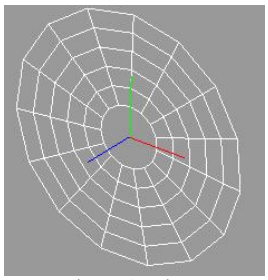
Figura 6. Disco

  - *Disco*: elemento que define una circunferencia plana a partir de un radio interno y un radio externo. Se define en el plano XY y centrado en el origen (figura 6).
  - *Disco Parcial*: elemento que define un segmento circular a partir de un radio interno, un radio externo, un ángulo de inicio y un ángulo de barrido. Se define sobre el plano XY y centrado en el origen (figura 7).
  - *Toroide*: elemento que define un toro a partir de un radio interno y un radio externo. El radio externo es el radio de la ruta circular que recorre la circunferencia de radio interno. Se encuentra sobre el plano XY y centrado en el origen (figura

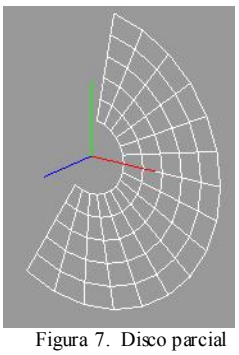
Figura 7. Disco parcial

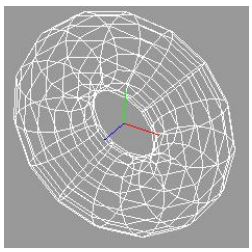
Figura 8. Toroide

  - Representación por la Frontera (Brep): elemento que contiene una representación geométrica a partir de un conjunto de vértices y un conjunto de caras triangulares que utilizan dichos vértices. Este tipo de representación se lee desde un archivo con formato Wavefront OBJ. El centro del objeto y su orientación, se encuentra definida en el archivo por la disposición de sus puntos (figura 9).

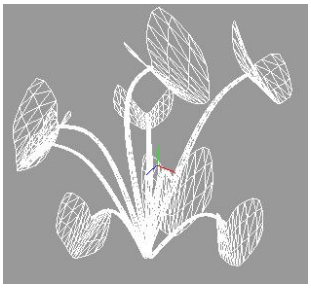
Figura 9. Representación por la Frontera.

- *Transformación*: nodo que representa una transformación geométrica sobre un nodo, ya sea un elemento o un mundo. Inicialmente se representan las tres transformaciones geométricas clásicas: traslación, rotación y escalamiento. Hay que tener en cuenta que la aplicación de transformaciones no es conmutativa, así que el orden en el que éstas se aplican es importante para el efecto que se quiere dar.
- *Grupo*: nodo de referencia que agrupa un conjunto de

elementos que forman una jerarquía local.

- *Referencia*: nodo que permite referenciar un nodo elemento o nodo grupo de un mundo con propósito de reutilización. Una referencia es un nodo atómico del cuál no se desprenden ni transformaciones ni hijos.
- *Material*: nodo que define las características de un material que se puede aplicar a un elemento como son: color especular (RGBA), color difuso (RGBA), color ambiente (RGBA), color de emisión (RGBA), y factor de brillo.

El diagrama de clases de la estructura que representa grafos de escena es (figura 10):

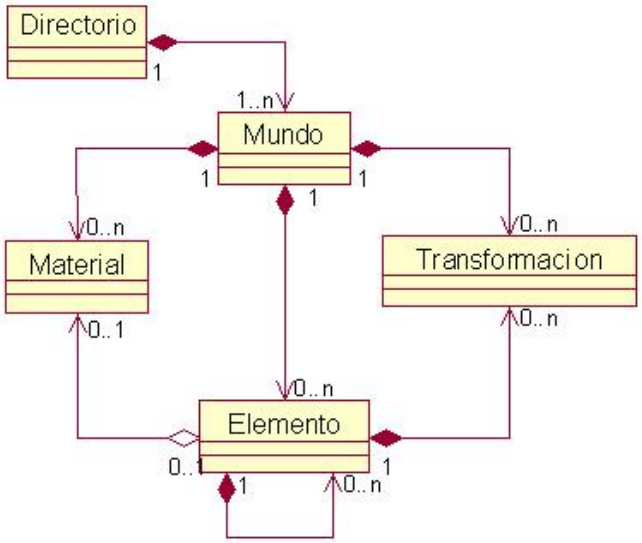

Figura 10. Diagrama de Clases de la Estructura

A *Vistas sobre la Representación*

Para la visualización de los mundos (grafos de escena) creados se definen tres tipos de vistas: tipo Árbol, tipo Grafo y modelo 3D.

1) *Vista tipo Árbol*: vista por defecto que muestra la jerarquía del objeto de interés con sus respectivos nodos. La estructura define el orden en el que el usuario construye el objeto de interés. En particular, una secuencia de transformaciones a aplicar sobre un elemento aparecen como hijos del elemento y se aplican de arriba hacia abajo. Esta representación utiliza la clase JTree de Java Swing (figura 11).

2) *Vista tipo Grafo*: Vista que muestra la estructura de grafo de escena clásico, en el que se destacan únicamente los nodos elemento y transformación. En este tipo de visualización no se muestran los materiales. La estructura define el orden a aplicar los nodos en un lenguaje de descripción geométrico. En particular, una secuencia de transformaciones a aplicar sobre un elemento aparecen antes del elemento y se aplican de abajo hacia arriba (figura 12). Esta representación utiliza el API JGraph [7].

Figura 11. Vista tipo Árbol del modelo Avatar

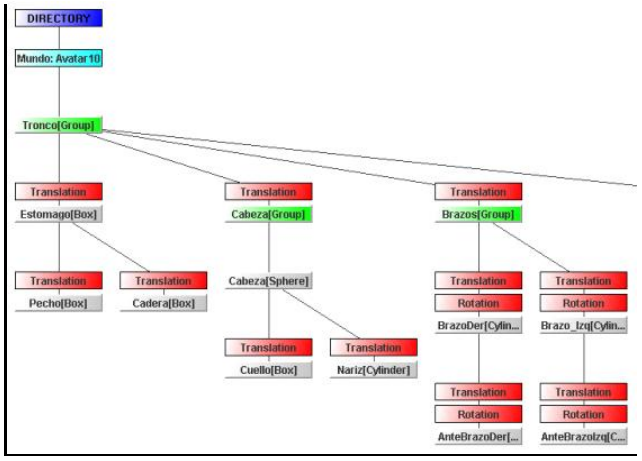

Figura 12. Vista tipo Grafo del modelo Avatar

3) *Vista modelo 3D*: Vista que muestra la interpretación del grafo de escena de un mundo utilizando el API JOGL [8] de descripción geométrica para obtener el modelo geométrico 3D resultante (figura 13). Este tipo de vista tiene varios parámetros que se pueden ajustar para mejorar la visualización (figura 14). Un modelo se puede ver con su representación en solo puntos, líneas o relleno, y con un sombreado plano o suavizado (Gouraud) siempre y cuando la iluminación se encuentre habilitada. También se puede activar o desactivar la grilla de referencia. Cada uno de los elementos se muestra con su eje de referencia local.

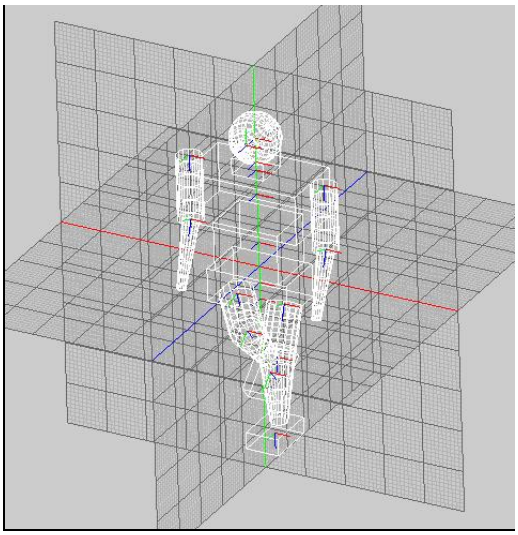

Figura 13. Vista modelo 3D del modelo Avatar

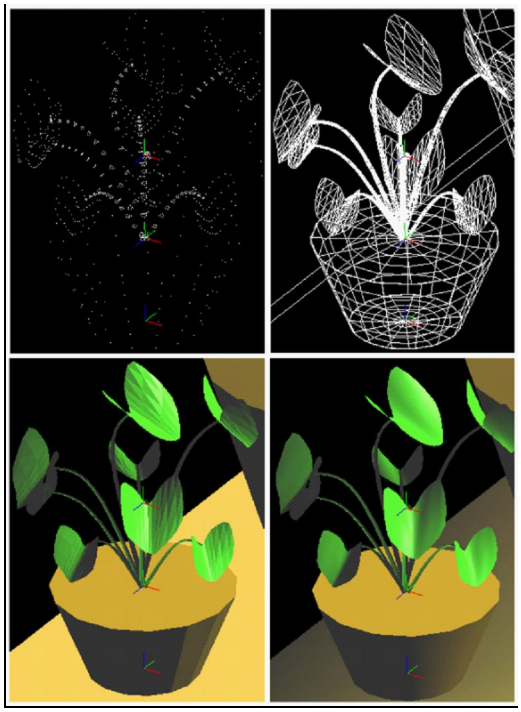

Figura 14. Diferentes vistas del modelo 3D (puntos, líneas, sombreado plano, sombreado suavizado) de objeto geométrico complejo.

B. Exportación a Lenguajes de Descripción Geométrica

Los modelos generados se pueden exportar a representaciones geométricas bajo JOGL (en lenguaje Java) [8] u OpenGL (en lenguaje C) [12][12]. Estas representaciones se pueden utilizar en aplicaciones gráficas desarrolladas utilizando JOGL u OpenGL respectivamente. Adicionalmente un modelo geométrico se puede exportar al lenguaje VRML 2.0 [1][14] soportado por diferentes herramientas de visualización (incluyendo navegadores de Internet).

C. Persistencia

Para la persistencia de la aplicación se utiliza una plantilla genérica que describe las partes que contiene la configuración de uno o varios mundos (grafos de escena). Se utiliza lenguaje XML para su representación con la siguiente configuración (figuras 15, 16 y 17).

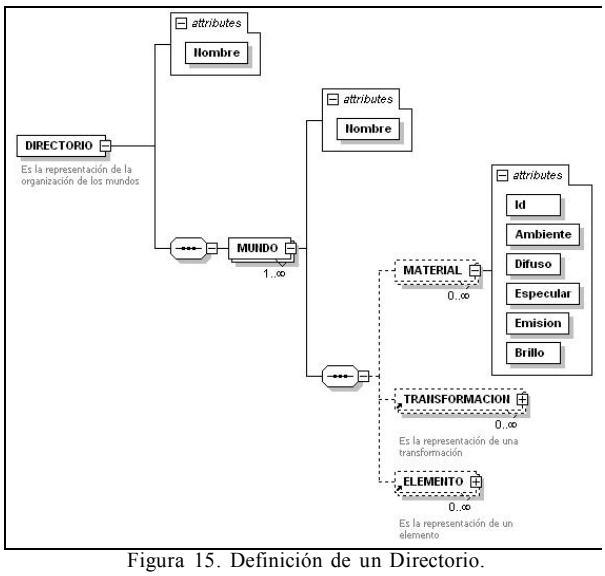

Figura 15. Definición de un Directorio.

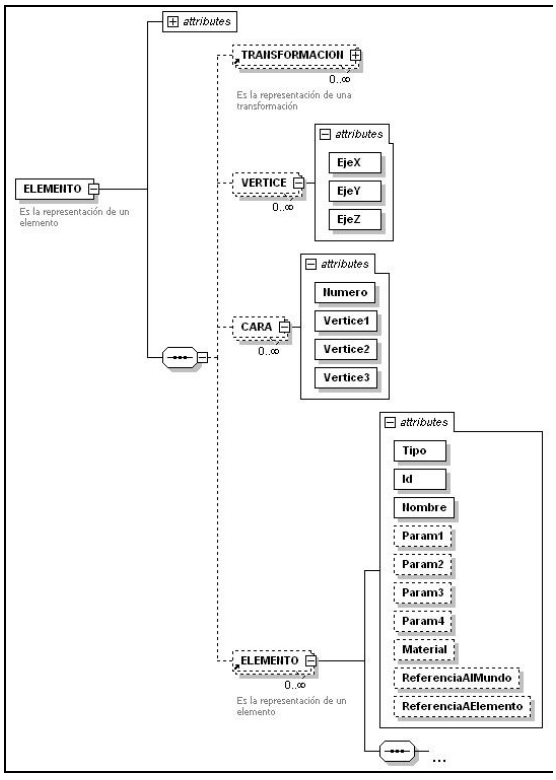

Figura 16. Definición de un Elemento.

Para los objetos geométricos bajo representación por la frontera, se utiliza un archivo formato Wavefront OBJ en donde cada línea del archivo tiene un encabezado compuesto por una

letra seguido de tres parámetros. Se define un archivo por objeto geométrico. Para los vértices el encabezado inicia con una *v* seguido de tres números flotantes los cuales definen las coordenadas X, Y y Z del vértice. Para las caras triangulares, el encabezado inicia con una *f* seguido de tres números enteros que definen los índices de los vértices que componen la cara. Cabe anotar que todas las caras son triángulos para garantizar que cada cara se define en un mismo plano.

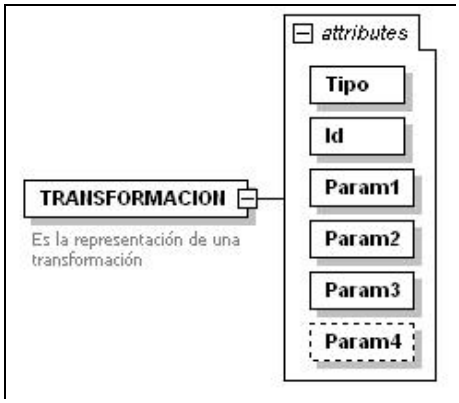

Figura 17. Definición de una Transformación.

D. Tecnologías de apoyo al Desarrollo

Para la construcción de la herramienta se utilizó como base el lenguaje Java, versión Java SE 1.5.0 que incluye la librería *Swing* para el entorno gráfico de la aplicación. Otra de las ventajas de utilizar un lenguaje como Java es la portabilidad de la aplicación hacia otros sistemas operativos diferentes a Windows.

Para la visualización 3D se utilizó la librería JOGL [8] la cual hace parte del proyecto para migrar el API de la especificación OpenGL 2.0 [9][13] en lenguaje Java.

Para la visualización de los grafos de escena se utilizó la librería JGraph [7] gracias a la facilidad para la construcción de grafos y su integración a los componentes de *Swing*.

Para la exportación de modelos geométricos, se utilizó la especificación de JOGL, OpenGL y VRML 2.0 [1][14] debido a su amplio uso y facilidad de visualización.

IV. RESULTADOS

Se crearon varios modelos geométricos jerárquicos para mostrar la capacidad de la aplicación en la construcción de objetos complejos y la exportación de los mismos. La primera prueba construye un modelo avatar utilizando únicamente elementos predefinidos (figura 18). En el lado izquierdo se tiene la vista modelo 3D generada por la aplicación, y en el lado derecho vemos el avatar exportado a VRML utilizando como visor Cortona [3].

Una segunda prueba construye un modelo de Bart utilizando únicamente elementos de representación por la frontera (figura 19). La figura de la izquierda muestra la vista modelo 3D de la aplicación en modo suavizado y la de la derecha es su exportación a VRML.

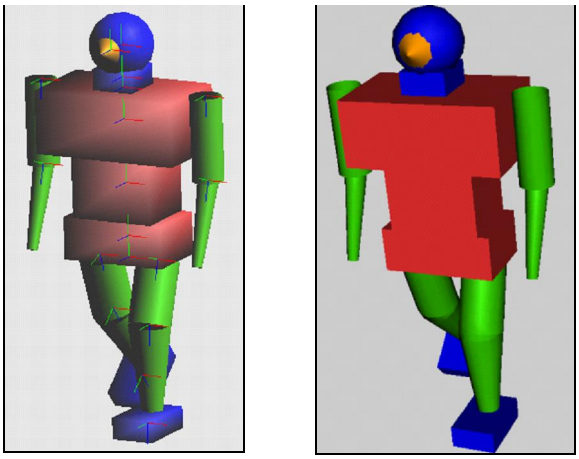

Figura 18. Prueba con elementos predefinidos:
vista modelo 3D (izquierda) y visualización en visor VRML (derecha).

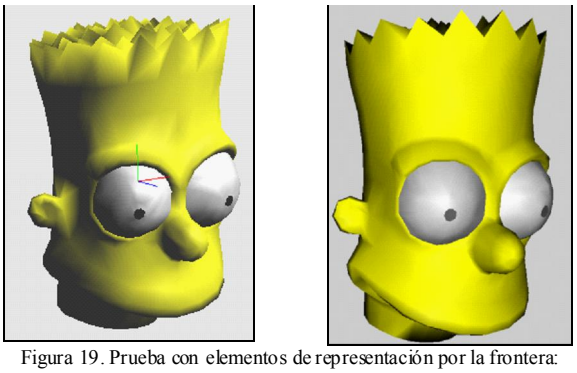

Figura 19. Prueba con elementos de representación por la frontera:
vista modelo 3D (izquierda) y visualización en visor VRML (derecha).

Por último, un modelo geométrico construido a partir de elementos predefinidos y de representación por la frontera (figura 20).

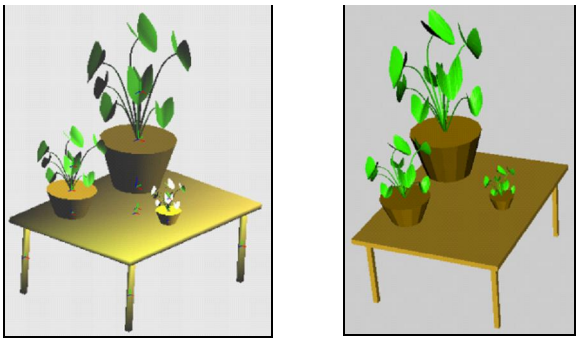

Figura 20. Prueba mixta integrando elementos predefinidos y elementos de representación por la frontera:
vista modelo 3D (izquierda) y visualización en visor VRML (derecha).

V. CONCLUSIONES Y TRABAJO FUTURO

La utilización de ayudas computacionales en el ámbito educativo es un campo que debe ser más explotado, ya que brinda un gran apoyo a los estudiantes y puede llegar a disminuir considerablemente la curva de aprendizaje de ciertos temas.

La utilización de herramientas de este tipo, ayuda a implementar en el ámbito educativo un sistema de "*learning by doing*" en donde los estudiantes se ven estimulados al observar los resultados de sus experiencias instantáneamente.

La aplicación de ayudas computacionales puede servir para la generación de modelos virtuales de aprendizaje soportados sobre plataformas en Internet para su expansión y uso global.

El trabajo puede continuarse en varios aspectos que complementan los resultados obtenidos y que ilustran la aplicación de otros temas de interés en computación gráfica. En particular se planea trabajar en las siguientes extensiones:

- Pruebas de usabilidad con un grupo de estudiantes de un curso de Computación Gráfica.
- Poder modificar el nivel de detalle de las primitivas geométricas predefinidas.
- Enriquecer el grafo de escena con: nodos que definan nuevas primitivas geométricas; nodos tipo cámara que definen las características del *viewport* y del observador; nodos relacionados con el modelo de iluminación que afectan la calidad visual (fuentes de luz); nodos de materiales a partir de texturas.
- Agregar la característica de movimiento (animación) en elementos que sean dinámicos.
- Implementar módulos de exportación a otros lenguajes de descripción geométrica.
- Implementar un modelo de iluminación global tipo *raytracing* para comparar contra el modelo local simplificado utilizado por eficiencia.
- Extender la interacción del usuario en los modelos construidos.

REFERENCIAS

[1] Ames, Andrea; Nadeau, David; Moreland, John. VRML 2.0 *Sourcebook (2nd edition)*. John Wiley & Sons, Inc. 1997.

[2] Bouvier, Dennis J. *Assignment: Scene Graphs in Computer Graphics Courses*. ACM SIGGRAPH, 2002.

[3] Cortona VRML Viewer Client 5.1., ParallelGraphics. http://www.parallelgraphics.com/products/cortona/ (último acceso 19 Dic. 2007).

[4] Cunningham, Steve; Bailey; Michael. *Lessons from scene graphs: using scene graphs to teach hierarchical modeling*. Computers & Graphics, No 25, pp 703 - 711, Elsevier Science Ltd., 2001.

[5] Exploratories. Brown University. *http://www.cs.brown.edu/exploratories/freeSoftware/catalogs/scenegraphs.html* (último acceso 19 Dic. 2007).

[6] Foley, James; Van Dam, Andries; Feiner, Steven; Hughes, John; Phillips, Richard. *Introduction to Computer Graphics*. Addison-Wesley, 1994.

[7] Java Graph Visualization and Layout (JGraph API), JGraph Ltd. http://www.jgraph.com (último acceso 19 Dic. 2007).

[8] JOGL API Project, Sun Microsystems. https://jogl.dev.java.net/ (último acceso 19 Dic. 2007).

[9] OpenGL. The Industry Standard for High Performance Graphics. http://www.opengl.org/ (último acceso 19 Dic. 2007).

[10] OpenSG. http://opensg.vrsource.org/trac (último acceso 19 Dic. 2007).

[11] OpenSceneGraph. http://www.openscenegraph.org/projects/osg (último acceso 19 Dic. 2007).

[12] Shreiner, Dave. *An Interactive Introduction to OpenGL Programming*. SIGGRAPH 2004.

[13] Shreiner, Dave; Woo, Mason; Neider, Jackie; Davis, Tom. *OpenGL(R) Programming Guide: The Official Guide to Learning OpenGL (5th edition)*. Addison-Wesley. 2005.

[14] VRML Virtual Reality Modeling Language, W3C. http://www.w3.org/MarkUp/VRML/ (último acceso 19 Dic. 2007).

Juan C. Ibarra López. Recibió el título de Ingeniero de Sistemas y Computación de la Universidad de los Andes (Bogotá, Colombia) en el 2007. Actualmente se encuentra estudiando Maestría en Ingeniería de Sistemas en la Universidad de los Andes, Bogotá, Colombia. Su campo de Investigación es el de Computación Visual más específicamente el de ambientes inmersivos de aprendizaje.
A partir de septiembre de 2007 es asistente Graduado en el proyecto de Investigación "Diseño y construcción de un sistema de visualización inmersiva, escalable y de bajo costo aplicado al entrenamiento y al entretenimiento educativo basados en imágenes y modelos 3D de órganos humanos" financiado por Colciencias - Instituto Colombiano para el Desarrollo de la Ciencia y la Tecnología "Francisco José de Caldas" en la Universidad de los Andes, Bogotá, Colombia.

Fernando De la Rosa Rosero. Se graduó como Ingeniero de Sistemas y Computación de la Universidad de los Andes (Bogotá, Colombia) en Abril de 1989. Como estudios de posgrado obtuvo una Maestría de Ingeniería de Sistemas y Computación en la Universidad de los Andes en Septiembre de 1991, un Diplôme d´Etudes Approfondies (D.E.A.) del Institut National Polytechnique de Grenoble (I.N.P.G), Francia, en Junio de 1992 y el Doctorado en Informática del Institut National Polytechnique de Grenoble (I.N.P.G.), Francia, en Noviembre 1996.
Fernando De la Rosa está vinculado al Departamento de Ingeniería de Sistemas y Computación de la Universidad de los Andes (Bogotá, Colombia) como profesor investigador desde 1997. Actualmente es Profesor Asociado y es miembro del grupo de investigación IMAGINE (Informática Gráfica) donde trabaja en las líneas de investigación de Robótica y Computación Gráfica.

UNIVERSIDAD
NACIONAL
DE COLOMBIA

SEDE MEDELLÍN

FACULTAD DE MINAS

120 años

TRABAJO Y RECTITUD