Docencia Asistida por Ordenador en Infografía.

F. Perales y C. Yániz

Dpto. de Matemáticas e Informática.
Universitat de les Illes Balears.
07071 Palma de Mallorca. España.
e-mail: dmifpl0@ps.uib.es, carlos@anim.uib.es

Resumen

La docencia asistida por ordenador se aplica a diferentes y muy diversas áreas o materias. En nuestro caso, hemos elegido los gráficos por ordenador por ser un tema donde inherentemente aparecen conceptos gráficos e interactivos. El objetivo de este trabajo es desarrollar un entorno donde el alumno pueda acceder a la gran mayoría de conceptos básicos relacionados con la Infografía. Está orientado a un primer curso de Informática Gráfica de una Ingeniería Técnica. Como objetivo complementario, el proyecto pretende, además de suministrar la información fundamental para el buen desarrollo de la asignatura, orientar al estudiante y ayudar al docente en dicha materia para una posterior y mayor formación. Por ello, se incluyen direcciones de interés en Internet, bibliografía complementaria y específica, asociaciones relacionadas, software shareware, congresos y conferencias. La utilización del lenguaje Java[1] hace posible una interactividad completa. Finalmente, otro objetivo fundamental es la ampliación de las librerías de Java para incluir nuevos métodos gráficos tridimensionales.

1. Introducción

Hoy en día es obvio el interés por parte de los investigadores y los docentes en la utilización masiva del World Wide Web (WWW). En nuestro caso el interés se centra en la utilización de esta herramienta en la docencia y educación. Nuestro énfasis en la docencia asistida por ordenador (CBL Computer Based Learning) se orienta en mejorar la comunicación entre el emisor y el receptor, es decir entre el educador y el alumno.

La red WWW se ha consolidado como un recurso muy valioso en el proceso de aprendizaje y educación. Como cualquier herramienta tiene sus ventajas y desventajas. Entre las principales características positivas podemos resaltar:

- la facilidad de exponer a un amplio número de individuos una información variada con un coste mínimo.
- esta información está generalmente disponible independientemente de la plataforma utilizada por medio de los estándares de publicación.
- es fácilmente actualizable, accesible y en muchos casos gratuita.
- la información puede llegar actualizada a zonas geográficas remotas. Se facilita enormemente la enseñanza a distancia.

En los aspectos negativos cabe destacar que la información es básicamente estática (libro electrónico) y limitada a nivel de interactividad. Esto se debe a que la

[1] Java es una marca registrada de Sun Microsystems Inc.

red WWW está constituida en su mayoría por páginas escritas en el lenguaje de hipertexto HTML. Las páginas que pueden contener textos y gráficos se interconectan mediante enlaces de hipertexto. En principio la única interactividad posible es pasar de una página a otra.

Por ello, el objetivo de este proyecto es precisamente buscar los mecanismos actuales que permitan integrar esta interactividad en el proceso de aprendizaje, manteniendo las ventajas originales. Es necesario ampliar por un lado el concepto de interactividad de tal manera que se modifique el concepto básico de libro electrónico y por otro el entorno de aprendizaje de forma que sea transparente y ampliamente interactivo.

Teniendo en cuenta esto, también hemos de considerar cuales son los requerimientos básicos que todo curso de educación superior por ordenador necesita para cubrir adecuadamente sus objetivos. Entre ellos podemos considerar:

- gran nivel de interactividad
- posibilidad de simulaciones
- secuencias animadas de determinados procesos
- respuesta rápida o inmediata y feedback específico
- entorno de aprendizaje agradable
- capacidad de adaptación
- facilidad de comunicación entre estudiante y profesor
- facilidad de actualización y utilización
- capacidad de motivación y orientación para una mayor especialización.

En la actualidad la utilización del lenguaje Java permite cubrir la gran mayoría de estos objetivos planteados a un coste razonable. La preparación de cursos de un alto nivel educativo mediante CBL requiere una gran inversión en cuanto a material humano y técnico. Es por ello que con la aparición del HTML, se sacrifica la calidad del producto final en beneficio de una reducción drástica del proceso de conversión de la información en procesadores de texto a la información accesible en WWW. Con Java se pretende buscar una solución que permita mejorar la calidad del producto final pero manteniendo un coste de desarrollo razonable. Evidentemente el proceso se complica y Java posee múltiples características positivas aunque la simplicidad no es una de ellas.

El resto del artículo se organiza de la forma siguiente: el apartado segundo hace referencia a trabajos anteriores relacionados con docencia en infografía. El apartado tercero incluye los principales temas incluidos en este curso de informática gráfica, describiendo que es lo desarrollado hasta el momento. En el apartado cuarto se introduce muy brevemente las ventajas del lenguaje Java. En los apartados quinto y sexto se analiza la estructura de un applet de Java típico y se explica como ampliar la funcionalidad de las librerías gráficas estándar de Java. Finalmente, el apartado séptimo, resume las principales conclusiones. En apéndice se puede ver un listado comentado de un applet y varias fotos de pantalla de la aplicación.

2. Trabajos anteriores.

HyperGraph [1] fue uno de los primeros trabajos de docencia en infografía a través de la WWW. Sólo contiene páginas HTML e imágenes en formato GIF por lo que no soluciona los inconvenientes de falta de interactividad planteados anteriormente.

Palmer y Helm proponen en [2] incorporar en las páginas HTML applets de

Java dentro de un curso de gráficos 3D por ordenador. Los applets son segmentos de código precompilado interpretados dentro del entorno WWW son los que nos permiten interactuar.

Partiendo de estos primeros trabajos realizados y en los cuáles la interactividad era reducida o el temario era restringido nosotros proponemos el entorno GrafiNet, plenamente interactivo formado por un conjunto integrado de teoría, problemas y prácticas interactivas que cubren todo el área de informática gráfica. Además proponemos la creación de un conjunto nuevo de librerías gráficas que extienden las funcionalidades de las librerías estándar de Java.

3. Contenido actual de GrafiNet.

Introducción a la infografía
Hardware en infografía
- Sistemas de visualización
- Dispositivos de visualización
- Otros dispositivos

Transformaciones de visualización
Algoritmos fundamentales
- Punto
- Líneas *
- Curvas *
- Relleno de polígonos
- Recorte *

Transformaciones geométricas
- Traslación
- Escalado
- Rotación
- Deformación
- Reflexión

Modelización *
Realismo
- Iluminación
- Color *
- Ray Tracing
- Aliasing

Matemáticas para infografía

Los temas marcados con un * son los que tienen algún applet con el que poder experimentar la aplicación de la teoría.

Actualmente se está trabajando las secciones de superficies y transformaciones 3D.

4. Breve introducción al entorno Java.

Java es probablemente uno de los avances más significativos en los entornos de software en los últimos años. Si el lenguaje de hipertexto HTML supuso la explosión de Internet mediante la edición y publicación de páginas estáticas de información, el lenguaje de programación Java supone además la inclusión de contenidos interactivos dentro de esas páginas que conforman la World Wide Web.

Tres elementos se combinan para hacer posible esta nueva revolución:

- Universalidad: Las aplicaciones Java son independientes de la plataforma. Cualquiera puede ejecutar los applets, que son aplicaciones pequeñas, seguras, dinámicas, multi-plataforma y en red. Los applets se pueden distribuir por Internet de manera segura, el usuario los carga a través de navegadores como Netscape o Microsoft IE.

- Potencia: Java es un lenguaje de programación orientado a objetos, poderoso, completo y fiable.

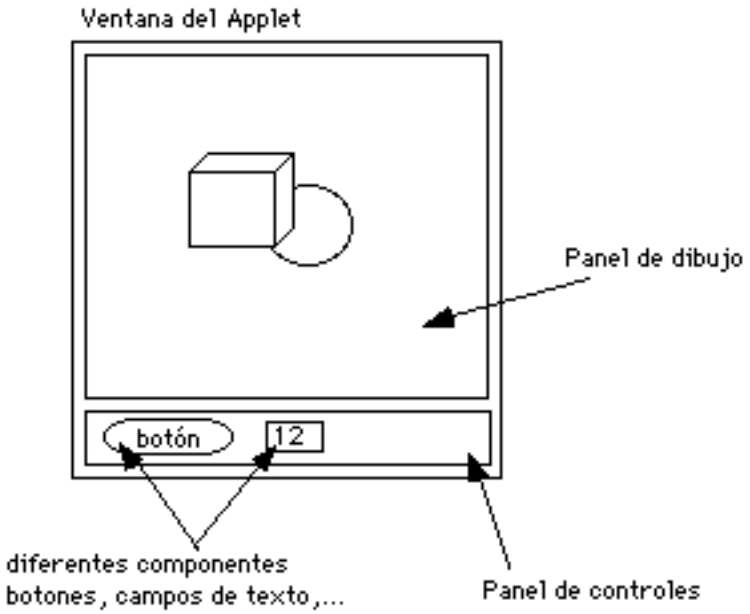
- Riqueza: Java es un conjunto rico de clases de objeto que proporcionan al programador abstracciones claras para muchas funciones de sistema habituales. Además de las clases estándar, el programador puede crear y distribuir nuevas clases organizadas en módulos independientes denominados paquetes.

Los entornos de programación que hemos utilizado para el desarrollo de GrafiNet son:
- Metrowerks Codewarrior 11 para Macintosh y para PC.
- Sun Java Developers Kit [4], que incluye la documentación completa en formato HTML de la API de Java versión 1.0.2.

A veces los applets tienen pequeñas diferencias de un intérprete a otro. Suelen ser problemas de visualización o refresco de pantalla, no de funcionamiento. Para evitar esto los applets son testados con diversos intérpretes de Java (Sun Java, Metrowerks Java, Netscape, Microsoft IE) sobre el máximo número de plataformas posible (Macintosh, PC, Sun y Silicon Graphics).

5. Estructura de un applet simple

Un applet típico tiene la siguiente forma:



El código correspondiente a esta estructura puede verse en el apéndice.

Aunque Java no es un lenguaje simple, una vez que se tiene implementada esta estructura es bastante fácil crear nuevos applets basados en ella. Ello permite

centrarse casi exclusivamente en el algoritmo gráfico que se desea mostrar.

Los applets hacen uso del paquete estándar java.awt que forma parte de la API 1.0.2. AWT son las siglas de Abstract Window Toolkit, es decir, equipo de herramientas de ventana abstracta. Este paquete incluye clases especializadas en:

- diseño de la interface: paneles de control, botones, campos de texto, ...etc...
- gestión de eventos: control del ratón y del teclado
- gráficos en 2D

Concretamente entre las clases para la creación de gráficos en 2D, la clase java.awt.Graphics contiene los métodos gráficos básicos como por ejemplo:

- drawLine: dibuja una línea entre dos puntos
- drawOval: dibuja una curva
- drawPolygon: dibuja un polígono
- drawRect: dibuja un rectángulo vacío
- fillRect: dibujo un rectángulo lleno
- getColor / setColor: controla el color

A la hora de implementar algunos algoritmos gráficos, y sobre todo si se trata de gráficos en 3D, las clases y métodos del paquete java.awt se han revelado insuficientes. Podríamos llamar a métodos nativos en otros lenguajes como C, pero esto violaría una de las normas de seguridad de los applets: no se puede tener acceso a recursos fuera del applet. La solución es por lo tanto escribir las clases y métodos que faltan usando las funciones básicas existentes. Además Java dispone de un mecanismo simple para crear nuestros propios paquetes y después poderlos incluir en otros proyectos. Por lo tanto en el siguiente apartado proponemos una extensión del paquete java.awt.

6. Java3D: un paquete de clases para gráficos 3D

En la sección de modelización se ha incluido un applet más completo cuya estructura es bastante más complicada que la de los otros applets. (Ver fotos en apéndice)

Queremos representar las funciones básicas de un modelador 3D:

- una cámara y una fuente de luz blanca
- dos primitivas: cubos y pirámides
- dos tipos de vista: con líneas o con polígonos coloreados
- posibilidad de rotación y zoom de la cámara
- en los objetos, posibilidad de traslación, rotación, escalado y deformación
- detección de caras ocultas en volúmenes convexos
- modelo de fronteras por compatibilidad con el formato de escenas 3D para Internet VRML [6].

En lo que respecta a la interface necesitamos incluir botones de control no sólo en la parte baja de la ventana, como en los applets simples, sino también a derecha e izquierda. Además es necesario tener una zona de ayuda "on-line" para explicar la funcionalidad de cada botón. El mensaje de ayuda aparece cuando el ratón pasa por encima del botón.

Las nuevas clases para gráficos 3D que hemos implementado forman el paquete java3d y son:

- Axes: representa 3 ejes que se pueden usar como referencia de la escena.
- Camera: una cámara tiene una posición y una orientación dentro de la

escena, que se definen por una matriz de transformación 4x4 con un algoritmo de quaternions. La cámara puede hacer un "render" de la escena desde su punto de vista.

- Cube: es una primitiva especial de 8 vértices y 12 caras triangulares
- Face3D: es una cara triangular definida por 3 puntos. Comparando su normal al eje de vista de una cámara sabemos si es visible o no. Si es visible calculamos su color con un sombreado de intensidad constante.
- Light: define una fuente de luz con un color y una orientación.
- Matrix: matriz de transformación para pasar de las coordenadas del mundo (escena) a las coordenadas de vista (cámara) y viceversa.
- Point3D: punto en 3D
- Primitive: primitiva para representar volúmenes. Consta de un cierto número de vértices y de aristas
- Pyramide: pirámide con 4 vértices y 4 caras triangulares
- Vector3D: vector en 3D

Un fichero en formato zip (java3d.zip) contiene todas estas clases. Se puede incluir el paquete java3d en otros proyectos siempre que sean de ámbito académico o de investigación. La dirección es la siguiente:

http://dmi.uib.es/people/carlos/java/Java3D

Una documentación completa de las clases y métodos se encuentra también en dicha dirección.

Asimismo confiamos en poder ampliar el paquete java3d y crear otros paquetes para gráficos, especializados por ejemplo en curvas o superficies. El mecanismo de paquetes de Java es abierto, permite que otros investigadores interesados en la enseñanza de la informática gráfica puedan añadir sus propios algoritmos.

7. Conclusiones y Trabajo Futuro

El estado actual del proyecto es "en desarrollo". La colaboración de profesores y estudiantes en el mismo ha sido altamente positiva y enriquecedora. Las diferentes perspectivas se juntan para un objetivo único: mejorar el proceso de aprendizaje e introducción de nuevas tecnologías y métodos de enseñanza. Al mismo tiempo, el alumno invierte un tiempo en la propia asignatura y temática de la materia lo que redunda en su propia formación durante el desarrollo del trabajo.

En la actualidad el número de módulos interactivos es reducido pero anualmente se irán incluyendo nuevos apartados. El lenguaje Java utilizado ha permitido introducir estos conceptos interactivos pero el coste no ha sido despreciable. A pesar de las ventajas que no se pueden negar a Java, hoy en día, el coste de desarrollo de CBL mediante este lenguaje se puede considerar todavía alto, aunque puede ser debido al aspecto de lenguaje de bajo nivel y a su estado embrionario. Por lo que, hace pensar que en un futuro próximo este coste decrezca.

Se puede acceder a GrafiNet en la siguiente dirección:

http://dmi.uib.es/people/paco/GrafiNet

Agradecimientos

En GrafiNet han colaborado estudiantes de tercero de Informática de Gestión y Telemática de la Universidad de Baleares:
F. Morote, J. M. Huéscar, J.C. Serra, J. L. Veny.

Bibliografía.

[1] HyperGraph - Teaching Computer Graphics Using Hypermedia (ACM Education Committee)

[2] CBL for graphics: the use of Java for tutorials on the Web, I.Palmer, P.Helm. Proceedings of the congress on 3D and Multimedia on the Internet, WWW and Networks. Bradford, UK. 16-18 april 1996.

[3] Manual de Java (The Java Handbook), P. Naughton. Osborne/McGraw-Hill, 1996.

[4] The Java Developers Kit: Version 1.0.2, Sun Microsystems Inc., 1996. http://java.sun.com

[5] Computer Graphics: principles and practise, J.D.Foley et al. Addison-Wesley, 1990.

[6] The VRML Sourcebook, Andrea L. Ames et al. Ed. John Wiley & Sons Inc, 1996.

Apéndice.

En las páginas siguientes incluimos:
- Listado de la estructura del código fuente de un applet simple
- Imagen del Sumario de GrafiNet, visto desde NetScape
- Imagen de la sección de Ray Tracing
- Imagen de la sección de Curvas 2D
- Imagen de la sección de Modelización

Listado de la estructura del código fuente de un applet simple

```
/**
 * Elementos que se deben dibujar.
 */
public class Circulo extends Canvas {
    // Definición y propiedades.....
}

public class cubo extends Canvas {
    // Definición y propiedades.....
}

/**
 * Esta clase representa una zona dónde de la pantalla dónde se va a dibujar.
 */
class PanelDibujo extends Panel {

    void nuevoCirculo() {
            // Añadir circulo .....
    }

    void nuevoCubo() {
            // Añadir cubo .....
    }

    void borrar() {
            // Borrar todo .....
    }

    void handleEvent() {
            // Gestión de eventos (Ratón, teclado) .....
    }
}

/**
 * Clase principal. Aqui empieza el programa.
 * Crea dos paneles, uno para la zona de dibujo, y otro para la zona de botones.
 */
public class AppletCirculos extends Applet {
    // Inicializar todos los elementos .....
    PanelDibujo panelDibujo = new PanelDibujo();
    Panel panelControl = new Panel();
}
```

Imagen 1: Sumario de GrafiNet, visto desde NetScape

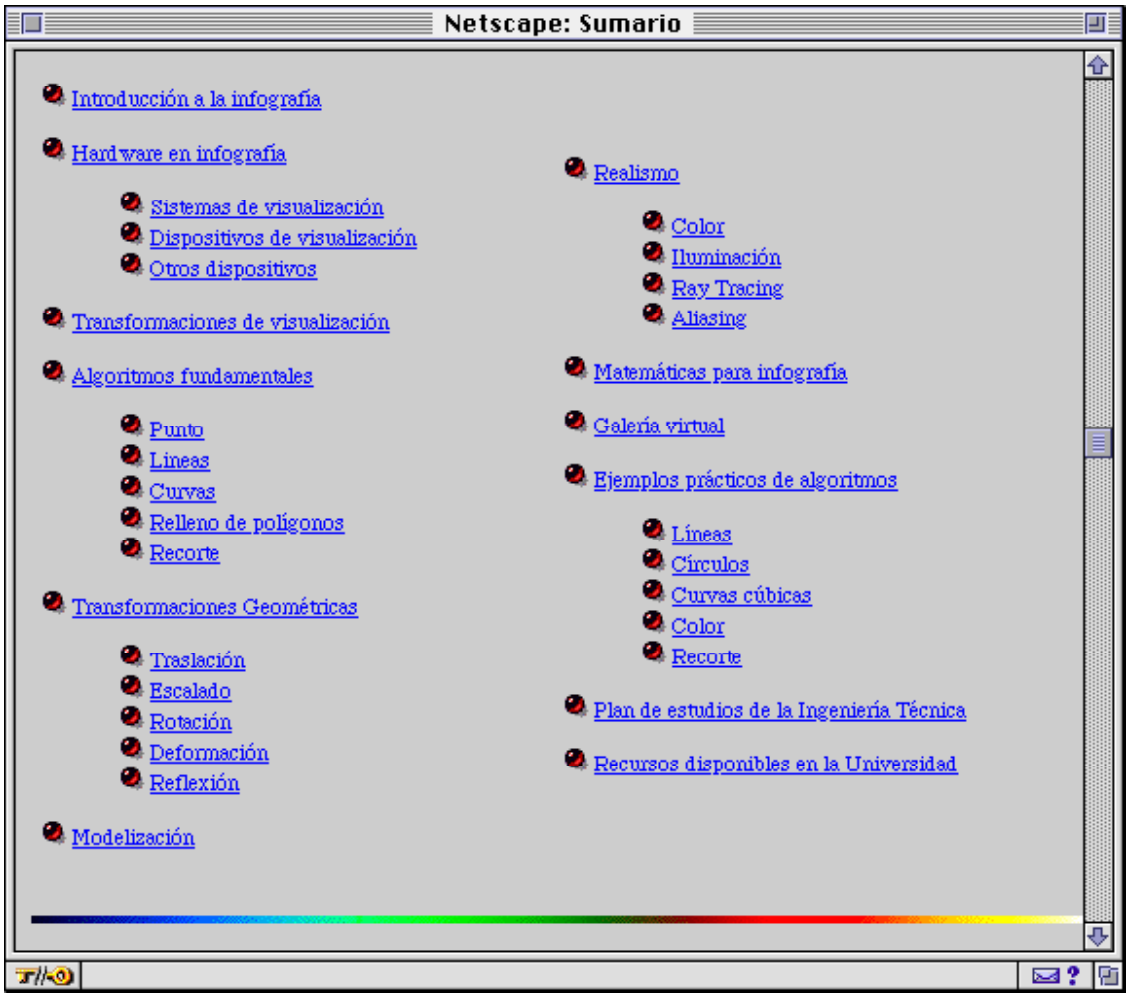


Imagen 2: Sección de Ray Tracing

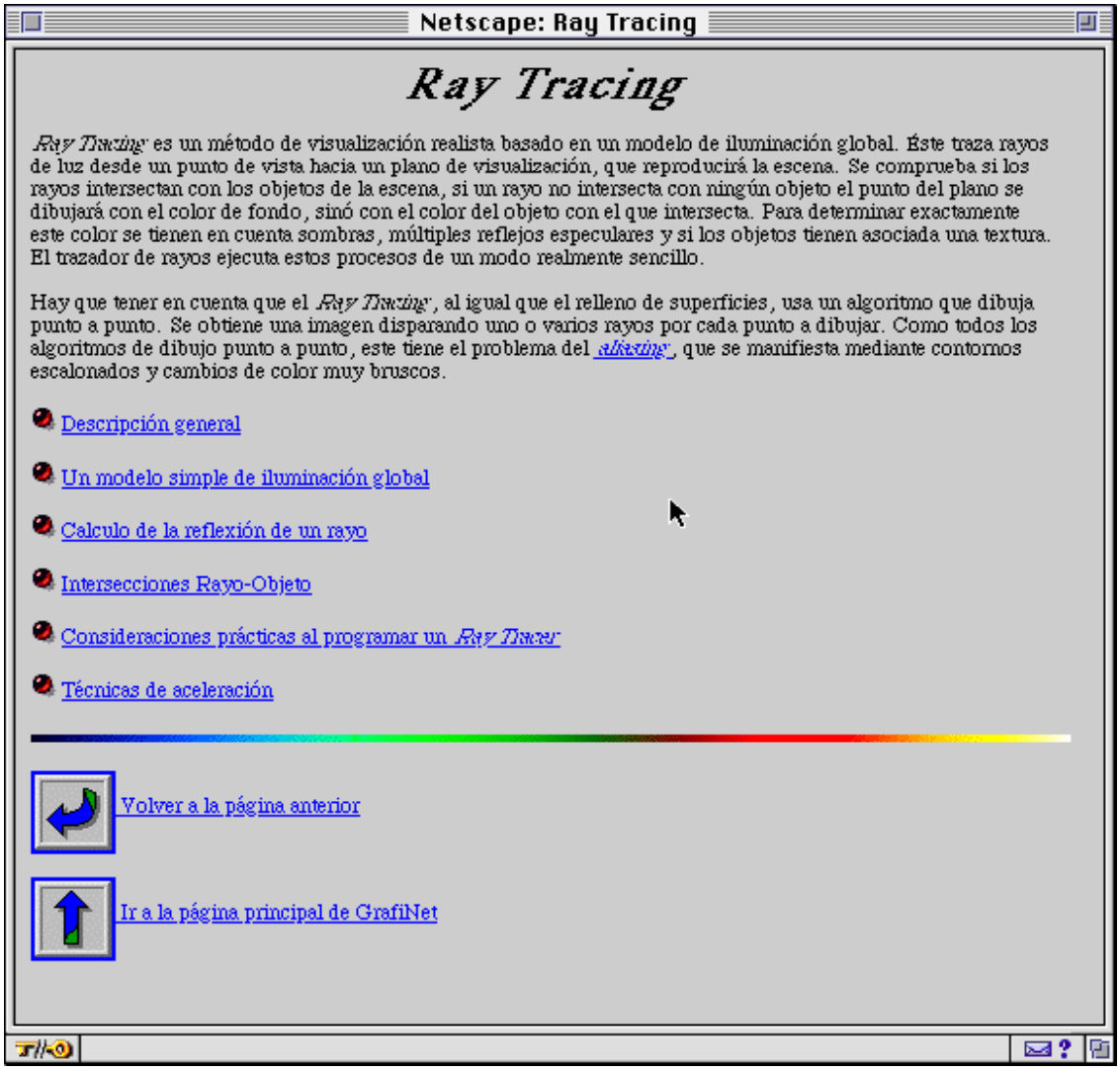


Imagen 3: Applet simple para ilustrar la sección de Curvas 2D

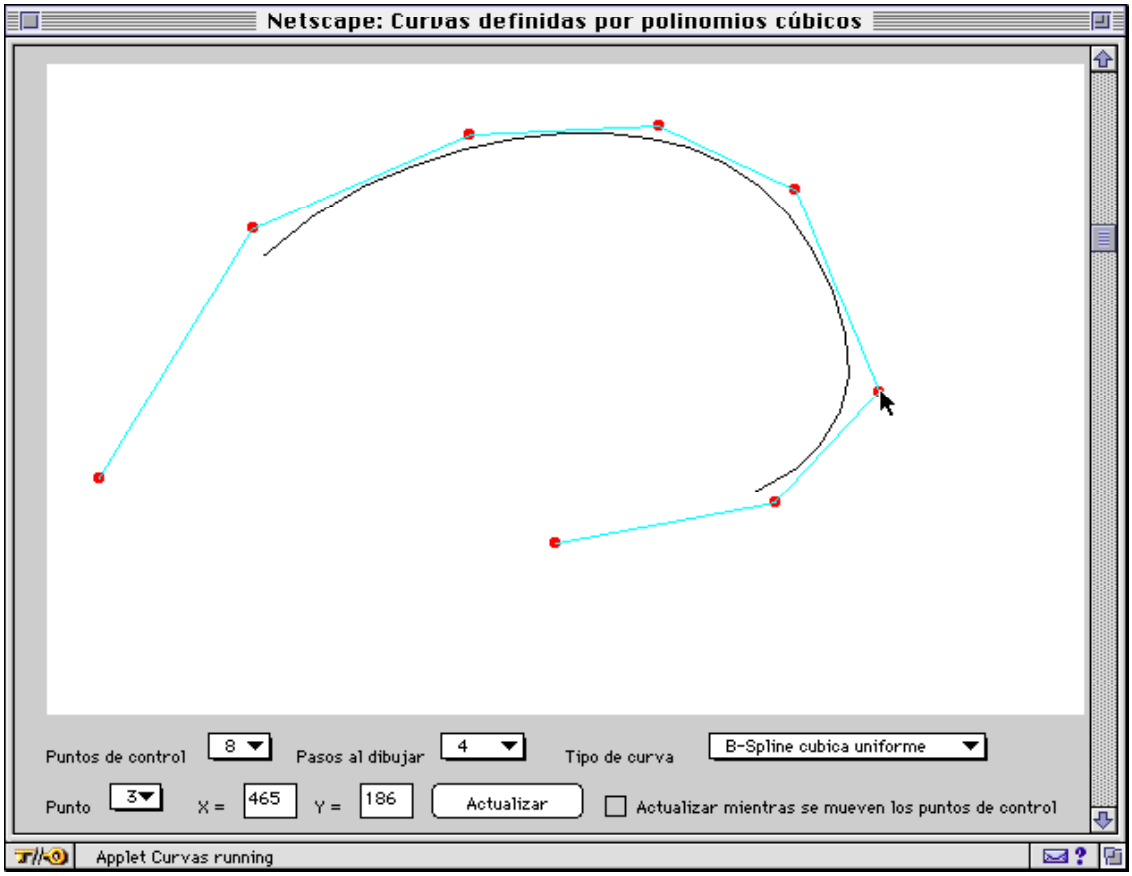


Imagen 4: Applet Java3D Modeler para ilustrar la sección de Modelización