Tipo de artículo: Artículo original

Visualizador avanzado de modelos 3D para la web

3D model's advanced viewer for the web

José Emilio Badia Valdés[1*], https://orcid.org/0000-0003-2189-1335

[1] Departamento de Informática, Facultad 4, Universidad de las Ciencias Informáticas. Km 2½, Autopista a San Antonio. La Habana.

*** Autor para correspondencia: jebadia@uci.cu**

Resumen

La generación, procesamiento y visualización de modelos tridimensionales es un estándar en cuanto a diseño o generación de gráficos por computadoras para numerosas ramas de la industria. En el presente trabajo se aborda el desarrollo de un visualizador Web avanzado de modelos tridimensionales para la Línea de Desarrollo de Videojuegos, del "Centro de Tecnologías Interactivas", de la Universidad de las Ciencias Informáticas (UCI), en, debido a que las opciones de visualización del visor existente, no cumplen con los estándares del centro y la distribución de las funcionalidades es deficiente. Para el desarrollo del visor, se utilizó la metodología XP, la tecnología WebGL y las bibliotecas Three.js, OrbitControl y Stats. El visor propuesto, permite cargar ficheros en los formatos OBJ, STL, FBX y glTF, desde los principales navegadores Web, comportándose entre los 50 y 60 FPS. Adicionalmente, ofrece funcionalidades para manipular el color de fondo de la escena, las luces en la escena, el movimiento de la cámara y las animaciones del modelo. Además de posibilitar la rotación automática del modelo y la activación del filtro Wireframe.

Palabras clave: 3D, CAD, CTI, Modelo, Software, Visualizador, Web.

Abstract

Generation, processing and visualization of 3D or three-dimensional models has nowadays become a standard in terms of computer graphics design for many branches of the industry. The present work is about the development of a three-dimensional model display tool for "Centro de Tecnologías Interactivas" of University of Computer Sciences (UCI), video-game's development branch, due the display option of the existing viewer doesn't fulfil the center's standards and the functionalities distributions. For the viewer's development, was used the XP methodology, WebGL technology and Three.js, OrbitControl and Stas libraries. The proposed viewer, allow the load of files with extension OBJ, STL, FBX y glTF in the main Web browsers, with a speed between 50 and 60 FPS. Additionally, it offers functionalities for the manipulation of: scene's background color, scene's light, camera's movement and the model's animation. Also, it allows the model's automatic rotation and the activation of the Wireframe filter.

Keywords: *3D, CAD, CTI, Model, Software, Display Tool, Web.*

Recibido: 10/01/2021
Aceptado: 19/02/2021

Introducción

Debido al avance de la tecnología y los gráficos generados por computadora, la visualización de objetos ha trascendido del plano 2D al mundo tridimensional, por lo que se ha convertido en una herramienta muy importante para los ingenieros, científicos y profesionales en general. El uso más natural que se le puede dar a los visualizadores es la simple funcionalidad de permitir ver una representación 2D o 3D de algún elemento; las tecnologías modernas permiten la interacción directa de los usuarios con los visores, permitiendo modificar sus características internas, ya sea por el posicionamiento de las imágenes o el color de la composición.

En la actualidad, el desarrollo de software en Cuba constituye uno de los eslabones principales en el proceso de informatización de la sociedad. El diseño gráfico se encuentra directamente relacionado con este campo. El procesamiento de imágenes y la representación gráfica resultan aspectos principales a tener en cuenta en el ámbito cinematográfico, así como en el desarrollo de videojuegos.

Con el avance de la ciencia, el desarrollo tecnológico produce artefactos cada vez más potentes, permitiendo generar imágenes tridimensionales capaces de lograr una similitud a la realidad con un alto nivel de precisión. El Centro de Tecnologías Interactivas (CTI), perteneciente a la Universidad de las Ciencias Informáticas (UCI), en la Línea de Desarrollo de Videojuegos, es uno de los centros del país que confecciona dichos objetos tridimensionales, con el objetivo de ser utilizados en los videojuegos que desarrollan.

Los trabajadores del CTI, en la Línea de Desarrollo de Videojuegos, comparten sus recursos o modelos 3D, de personajes y *assets* de manera manual o a partir del sistema de control de versiones que utilizan, para versionar el código de las aplicaciones desarrolladas. Recientemente se desplegó un sistema web para administrar modelos 3D de forma que se tenga acceso a los objetos diseñados para los videojuegos de manera centralizada.

Esta herramienta cuenta con un visor incorporado que permite visualizar objetos en formato OBJ y FBX, utilizados en el desarrollo de videojuegos. Sin embargo, las opciones de visualización, no cumple con las necesidades del centro, debido al mal uso del espacio de renderizado del modelo, la escasa interacción del usuario con el entorno y la deficiente distribución de las funcionalidades, dando lugar a un mal aprovechamiento de la tecnología y a una mala aceptación por parte de los diseñadores del centro. Además de no poseer la capacidad de leer archivos en formato Standard Triangle Language (STL, por sus siglas en inglés) y Graphics Library Transmission Format (glTF, por sus siglas en inglés), lo que está en contraposición con las nuevas necesidades del centro.

Materiales y métodos

A causa de las características del proyecto, es necesario utilizar una metodología de desarrollo ágil, que le permita a los especialistas reducir el tiempo de desarrollo, para su posterior explotación. Teniendo esto en cuenta, se decidió utilizar, dentro de las metodologías ágiles, XP, permitiendo cierta flexibilidad a lo largo del desarrollo del proyecto y una interacción directa con el usuario.

Para llevar a cabo de forma satisfactoria el desarrollo de la solución, se utilizará como editor de código el Visual Studio Code y el lenguaje de programación JavaScript, siendo así, el Xampp y el Mozilla Firefox los encargados de permitir la visualización de los modelos 3D en la web. Pero para lograr esta meta, es necesario utilizar una biblioteca especializada en el ámbito del desarrollo de escenas 3D en la web, la biblioteca seleccionada fue Three.js, permitiendo la carga de los diferentes elementos y la manipulación de los materiales y la estructura de los objetos tridimensionales. Los métodos de investigación utilizados en este trabajo fueron:

Análisis-Sintético: Permitió la asimilación de los fundamentos teóricos necesarios para enfrentar el proceso de desarrollo, el análisis de las tecnologías a utilizar, logrando así, obtener una síntesis de las principales características de las mismas.

Inducción-Deducción: Garantizó la generalización de las estructuras bases para la creación de visualizadores tridimensionales webs.

Entrevista: Admitió recopilar información y tener un mayor acercamiento a las necesidades del cliente a la hora de trabajar con la aplicación.

Resultados y discusión

El resultado desarrollado toma la forma de una aplicación web encargada, mediante la librería Three.js y el motor gráfico WebGL, de representar modelos tridimensionales en los formatos STL, FBX, OBJ y glTF. El visor tridimensional no se limita solo a la representación tridimensional, además permite la interacción con el modelo y el entorno representados. Mediante un panel desplegable, el usuario no solo es capaz de modificar las propiedades del entorno, ya sean las luces presentes o el fondo de escena, también tiene acceso a propiedades del modelo como son: animaciones, capa de filtro y rotación. Con la ayuda del mouse, el modelo puede ser rotado o amplificado/reducido (zoom). Antes de comenzar a describir las propiedades del software desarrollado, es necesario definir las características de las tecnologías utilizadas.

¿Qué es *WebGL*?

Es el estándar para gráficos 3D en la web, se diseñó con el propósito de renderizar gráficos 3D y gráficos interactivos 3D. Esta deriva de la biblioteca OpenGL ES 2.0 (Sistema embebido) y proporciona un buen rendimiento en modernos hardware 3D (Parisi, 2012), (Haines, Angel, 2018).

> "Es una API JavaScript que puede ser usada con HTML5, ya que, el código *WebGL* es escrito dentro de la etiqueta <canvas>. Esto es una especificación, que permite al navegador acceder a la GPU de la computadora que se esté utilizando".

Aun sabiendo el concepto general de *WebGL*, todavía no queda claro la función que realiza. Según la página oficial para desarrolladores de Mozilla, se emplea fundamentalmente en:

> "Mostrar aplicaciones en el navegador basadas en OpenGL para poder así, renderizar contenido 2D y 3D en una página HTML sin tener que utilizar *plug-ins*. Los programas de *WebGL* consisten en un conjunto de código JavaScript y *Shader* que se ejecutan mediante la (GPU) de la computadora".

Para la correcta visualización del modelo tridimensional, es necesario la ejecución de un complejo proceso para las computadoras, el cual demanda altos recursos de hardware, debido a los complejos cálculos matemáticos que son necesarios realizar para la obtención de un resultado satisfactorio. Dicho proceso es conocido bajo el nombre de renderizado.

Renderizado

> "Proceso de generar una imagen tomando como base un modelo mediante el uso de programas informáticos. En cuanto a los gráficos, una escena virtual es descrita usando información como: geometrías, texturas, luces, las cuales pasan a través de un programa de renderizado. El resultado de estos programas de renderizado sería la imagen digital"

Existen dos tipos de renderizado:

- Renderizado por Software: Todos los cálculos para renderizar están hechos por la Central Processing Unit (CPU, por sus siglas en inglés).
- Renderizado por Hardware: Todos los cálculos para renderizar son hechos por la GPU.

Para lograr renderizar un modelo tridimensional es necesario, utilizar un método de renderizado de los tres existentes, los métodos basados en cliente (utilizado por *WebGL*), los del lado de servidor y, por último, los métodos híbridos. Los métodos del lado del servidor tienen la ventaja de que el modelo es completamente renderizado en el servidor y la imagen resultante es transmitida al cliente. Este método es muy eficiente cuando los recursos hardware de los usuarios finales son limitados. La desventaja radica en la mala interacción en tiempo real con el modelo y un mayor retraso en el tiempo de respuesta del servidor. El renderizado local es realizado mediante la CPU o GPU de la computadora del

usuario, por lo que recibe el nombre de renderizado basado en cliente. La geometría es descargada en las computadoras de los clientes, la cual mediante el uso de los recursos hardware que disponga permitirá el posterior renderizado (Martin, 2000).

Three.js

Es una biblioteca JS y a su vez una API usada para la creación de animación 3D mediante el uso de la GPU en el navegador. Para el renderizado de los modelos creados utiliza el motor gráfico WebGL, el cual ya está incluido en todos los navegadores (Haines, Angel, 2017).

Dentro de las características principales están:

- Creación de escenas tridimensionales.
- Aplicación de efectos como el desenfoque.
- Inclusión y eliminación de objetos 3D en tiempo de ejecución.
- Interacción con los modelos 3D de la escena en tiempo de ejecución.
- Soporte para diferentes modelos básicos.
- Optimización para la construcción de modelos básicos.

Propiedades y estructura de la solución propuesta

Para el desarrollo del visor, como se mencionó anteriormente, se utilizó la biblioteca JavaScript Three.js, encargada de la lógica de representación de la escena. Todas las características de los elementos como: luces, fondo y modelos a representar, son manipulables a través de los métodos proporcionados por Three.js (Kenwright, 2019). Como lenguajes de maquetado se utilizaron: HTML5 junto al lenguaje de plantillas EJS y Materialize como librería CSS. Las peticiones del cliente se operaron mediante Node.js, un entorno en tiempo de ejecución y el *framework* Express, siendo el último, el encargado de convertir las peticiones *url* del usuario en datos legibles para la aplicación, este proceso se puede observar en la figura 1.

Figura 1. Modelo de funcionamiento de la petición.

Una de las funcionalidades más importantes del visor es la posibilidad de cargar elementos en los formatos OBJ, STL, FBX y glTF. Fue necesario crear un mecanismo, que, en tiempo de ejecución, permitiera representar cualquiera de estos formatos sin tener que realizar cambios en el código fuente. La solución óptima fue, el uso de los patrones de diseño, específicamente, el Abstract Factory (GOF, 1994), figura 2. El patrón seleccionado provee una interfaz para la creación de objetos en una superclase, pero permite que las subclases decidan el tipo de objeto que va a ser creado.

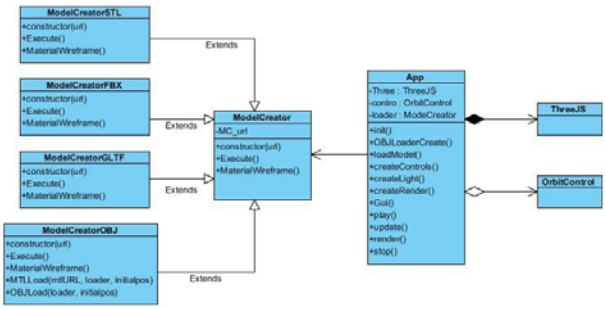

Figura 2. Patrón Abstract Factory en el visor.

Una característica que está arraigada a los visores tridimensionales, es la existencia de un área de visualización del modelo. Normalmente se encuentra ubicada en el centro del visor, para que sea lo primero que el usuario observe una vez que accede a la herramienta. La ventana de manipulación de modelos puede encontrarse en visores tridimensionales que además de representar el modelo, permitan la manipulación de los elementos dentro de la escena, un ejemplo de este tipo de visor, se encuentran en sitios webs como: Sketchfab o GooglePoly. Ubicada en el lado izquierdo de la pantalla y accedida mediante un botón en la esquina inferior derecha, posee las funciones principales para la manipulación de la escena y el modelo, ver figura 3. En ella se encuentran funcionalidades como: el movimiento del modelo por el espacio, manipulación de la animación, entre otras opciones. La presencia de un gráfico que representa la velocidad del visor en Fotogramas Por Segundo (FPS por sus siglas en ingles) y Milisegundos (MS por sus siglas en ingles), en la zona superior izquierda de la pantalla, se hace necesaria para comprobar el rendimiento en los distintos entornos de ejecución.

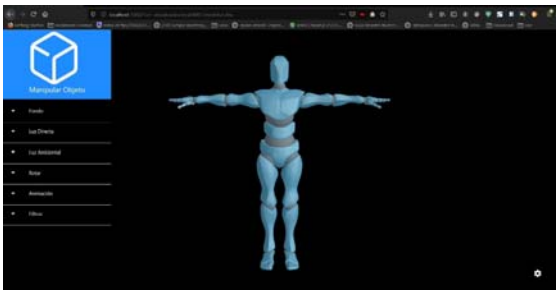

Figura 3. Menú de manipulación.

Prueba de rendimiento del software desarrollado

Utilizando el gráfico anteriormente mencionado, el visor fue sometido a una prueba de rendimiento, para observar su comportamiento en un entorno de trabajo similar al entorno de despliegue. En un entorno 3D, una velocidad de ejecución de 15 FPS, se considera tiempo interactivo, pero no real. Se consideran de tiempo real, aplicaciones con una velocidad de ejecución que oscilen entre 30 y 60 FPS. Mientras que, entornos que necesiten una interacción fluida con el modelo, debe ejecutarse a una velocidad de 50 FPS o superior (Atazadeh, 2019). Los equipos utilizados para la prueba de rendimiento poseían las siguientes características:

- Computadora1:
  - Procesador: Core I-3 5ta generación.
  - RAM: 8 GB.
- Computadora2:
  - Procesador: Core I-5 8va generación.
  - RAM: 8 GB.
- Computadora3:
  - Procesador: Core I-3 3ra generación.
  - RAM: 4 GB.
- Smartphone:
  - Procesador: Kirin 970.
  - RAM: 4 GB.
- Navegadores:
  - Mozilla Firefox.
  - Google Chrome.

Con los resultados de las pruebas, figura 4, se pudo llegar a la conclusión de que el visor tiene una media de velocidad de ejecución entre 50 a 60 FPS en los navegadores Firefox. Mientras que la ejecución en Google Chrome no surgió problemas hasta que, se observó una fluctuación de la velocidad de 45 a 55 FPS en modelos con estructura compleja como el de la imagen figura 5. La baja velocidad en Chrome se debe, al nivel de procesamiento necesario para la representación del modelo descrito anteriormente y el balance de carga con las aplicaciones que están ejecutándose. Por lo que se sugiere el uso de Firefox sobre Chrome debido a que el primero, en modelos complejos, brinda una interacción más fluida.

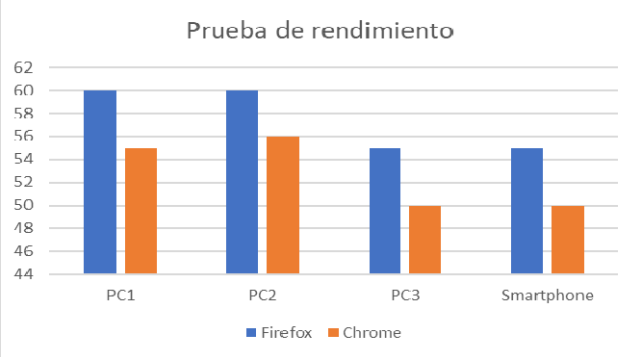

Figura 4. Resultados prueba de rendimiento

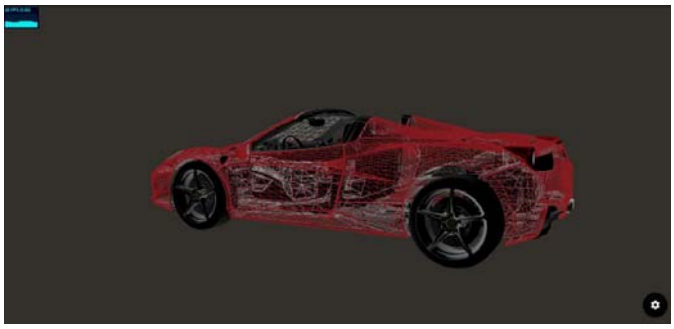

Figura 5. Representación del modelo formato glTF

Prueba de aceptación del software desarrollado

Antes del despliegue de una aplicación, es necesario realizar una serie de pruebas que validarán el cumplimiento de los requerimientos del usuario. El tipo de prueba a realizar depende de la metodología de desarrollo seleccionada, el visor tridimensional fue desarrollado siguiendo el marco de trabajo propuesto por la metodología ágil XP. Las pruebas de aceptación se realizan por cada una de las etapas que transite el desarrollo del proyecto, la no aprobación de una de estas pruebas permite la captura de las no conformidades del cliente con las funcionalidades desarrolladas, permitiendo la subsanación de los errores cometidos (Reyes, 2016). Las tablas 1,2,3,4 demuestran los resultados obtenidos en las diferentes etapas de desarrollo del visor.

Tabla 1: Prueba de aceptación HU1_P1

Caso de prueba de aceptación	
Código: HU1_P1	Historia de usuario: 1
Nombre: Cargar modelos	
Descripción: Prueba para comprobar el cumplimiento de la funcionalidad cargar modelos.	
Condiciones de ejecución: El usuario debe poseer la dirección física (dentro de la computadora) del modelo a cargar.	
Pasos de ejecución: El usuario debe indicar donde se encuentra el modelo a cargar para que, la dirección sea introducida manualmente en el código. Una vez introducida la dirección y reiniciado el servidor, el usuario deberá recargar en navegador para, obtener el nuevo modelo.	
Resultados esperados: El modelo se carga de forma satisfactoria.	

Tabla 2: Prueba de aceptación HU4_P1

Caso de prueba de aceptación	
Código: HU4_P1	Historia de usuario: 5
Nombre: Animación del modelo.	
Descripción: Manipular animación del modelo.	
Condiciones de ejecución: El modelo debe de haber sido cargado y renderizado en el navegador.	
Pasos de ejecución: El usuario selecciona la opción de animación en el menú de manipulación de modelos y ejecuta el comando destinado para esta funcionalidad.	
Resultados esperados: La animación se ejecuta correctamente.	

Tabla 3: Prueba de aceptación HU8_P1

Caso de prueba de aceptación	
Código: HU8_P1	Historia de usuario: 9
Nombre: Manipular movimiento del modelo.	
Descripción: Manipular movimiento del modelo en la escena.	
Condiciones de ejecución: El modelo debe de haber sido cargado y renderizado en el navegador.	
Pasos de ejecución: El usuario utiliza el mouse para rotar el modelo en la escena.	
Resultados esperados: El modelo rota según la dirección del mouse.	

Tabla 4: Prueba de aceptación HU12_P1

Caso de prueba de aceptación	
Código: HU12_P1	Historia de usuario: 13
Nombre: Cargado por URL.	
Descripción: Cargar modelo mediante URL.	
Condiciones de ejecución: El usuario debe poseer la dirección electrónica (URL) del modelo.	
Pasos de ejecución: El usuario debe proporcionar la dirección URL y ubicarla dentro de los parámetros requeridos.	
Resultados esperados: El modelo debe cargar correctamente.	

En total se realizaron 14 pruebas de aceptación, obteniendo como resultado; una no conformidad, solucionada posteriormente, y las 13 tareas aprobadas por el usuario. Sumando a la satisfacción del usuario con la construcción del software, la velocidad de ejecución en los principales navegadores oscila de 50 a 60 FPS; se afirma que el software desarrollado cumple con los estándares demandados por el usuario, por lo que se concluye que la solución desarrollada es adecuada para su explotación por los trabajadores del CTI.

Conclusiones

En este trabajo se desarrolló un visor avanzado de modelos 3D para la Web, para la Línea de Desarrollo de Videojuegos del CTI, que permite representar archivos en los formatos FBX, OBJ, STL y glTF. El visor propuesto, utiliza la tecnología WebGL y permite la representación en los principales navegadores Web. Adicionalmente se concluye:

- El soporte para cargar ficheros de los cuatro formatos incorporados, posibilita usar el visor propuesto como herramienta de apoyo en diversas áreas, sin comprometer el rendimiento, tales como: Impresión 3D, Diseño Industrial, Animación Digital y Desarrollo de Videojuegos.
- El visor desarrollado permitirá a los trabajadores del centro CTI, en la Línea de Desarrollo de Video- juegos, tener un software, al alcance de una URL, que les facilitará la centralización y visualización de antiguos, o recientes modelos, sin la necesidad de tener que descargarlos directamente en sus computadoras.
- El visor permite una interacción con el modelo a una velocidad de 50 a 60 FPS en los navegadores Web principales. Aunque, la velocidad de ejecución puede ser comprometida, debido a la complejidad de los modelos representados y al balance de carga de la computadora; por lo que se recomienda, el uso del navegador Firefox por la fluidez de las interacciones que brinda en estos casos.

Agradecimientos

A mis tutores por ayudarme con la estructura de la investigación y al CTI por permitir el uso de sus computadoras para la etapa de prueba.

Conflictos de intereses

El conflicto de intereses se encuentra ubicado en la selección del formato del modelo a representar; cada formato posee propiedades que, en ciertas circunstancias, lo pueden hacer resaltar sobre otro.

Contribución de los autores

- Investigación y Software: José Emilio Badia Valdés
- Investigación: Búsqueda de información sobre las tecnologías utilizadas para el desarrollo de un visor tridimensional web.
- Software: Desarrollo del visor tridimensional para la web.

Financiamiento

Desarrollo del software financiado por el autor con recursos propios. Etapa de prueba de rendimiento financiada por el CTI.

Referencias

Parisi, Tony. Chapter 1 An Introduction to WebGL. En: Mary Treseler. WebGL: up and running. Ciudad: Sebastopol, California. 2012, p 9 – 10

Ioana M. Martin. Adaptive Rendering of 3D Models Over Networks Using Multiple Modalities. [En línea]. IBM T.J.Watson Research Center, Research Report. 2000. [Consultado el: 15 de enero de 2020] 2 p. Disponible en: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.203.2569&rep=rep1&type=pdf

ED Angel, Eric Haines, and Dave Shreiner. Getting started with webGL and three.js. ACM SIGGRAPH 2018 Courses, 2018, Article 2: 1–82. DOI: https://doi.org/10.1145/3214834.3214861

Ed Angel, Eric Haines. An interactive Introduction to WebGL and Three.js. ACM SIGGRAPH, 2017, 14, p. 1 – 95.

The Gang of Four (GOF). Design Patterns: Elements of Reusable Object-Oriented Software. United States, Addison-Wesley, 1994. 395 p.

Br. Neldin Noel Pérez Reyes, Br. Sintya Milena Meléndez Valladarez, Br. Maria Elizabeth Gaitan. Metodologia Ágil De Desarrollo De Software Programacion Extrema. Tesis de maestría. Universidad Nacional Autonoma De Nicaragua, Managua, 2016.

Kenwright, B. Visualization with Three.js. En: 12th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia 2019, Brisbane, Australia.

Behnam Atazadeh, Abbas Rajabifard and Mohsen Kalantari. Assessing Performance of Three BIM-Based Views of Buildings for communication and Managemento of Vertically Stratified Legal Interests. International Journal of Geo-Information <<Research and Development Progress in 3D Cadastral Systems>>, 2019: 102 – 128.