2.0 MANUAL TÉCNICO:
CONFIGURACIÓN

1. Plataformas de Trabajo.

Para el desarrollo de nuestro ambiente jurásico fue necesario trabajar bajo el entorno de desarrollo integrado de Microsoft Visual Studio (IDE) el cual es utilizado para desarrollar programas informáticos, aplicaciones, y servicios, para nuestro caso, siguiendo de la mano las operaciones de OpenGL que se considera principalmente una API (una interfaz de programación de aplicaciones) el cual nos brinda un gran conjunto de funciones que podemos usar para manipular gráficos e imágenes. Sin embargo, OpenGL por sí mismo no es una API, sino simplemente una especificación, desarrollada y mantenida por el Grupo Khronos.

La especificación OpenGL define exactamente cuál debe ser el resultado/salida de cada función y cómo debe funcionar. Luego, depende de los desarrolladores como nosotros que implementemos estas especificaciones encontrar una solución de cómo debería operar esta función. Dado que la especificación de OpenGL no nos brinda detalles de implementación, las versiones desarrolladas reales de OpenGL pueden tener diferentes implementaciones, siempre que sus resultados cumplan con la especificación (y, por lo tanto, sean los mismos para los usuarios).

Es importante denotar que para el desarrollo de este proyecto jurásico se trabajó con las bibliotecas OpenGL que están escritas en C y permiten muchas derivaciones en otros lenguajes, pero en esencia sigue siendo una biblioteca C. Dado que muchas de las construcciones de lenguaje de C no se traducen tan bien a otros lenguajes de nivel superior, OpenGL se desarrolló con varias abstracciones en mente. Una de esas abstracciones son los objetos en OpenGL.

Se definieron múltiples objetos para este proyecto de recreación jurásica, siguiendo las especificaciones de requerimientos funcionales del apartado anterior. Un objeto en OpenGL es una colección de opciones que representa un subconjunto del estado de OpenGL. Por ejemplo, podríamos tener un objeto que represente la configuración de la ventana de dibujo; luego podríamos establecer su tamaño, cuántos colores admite, etc. Uno podría visualizar un objeto como una estructura tipo C.

Lo bueno de usar estos objetos es que podemos definir más de un objeto en nuestra aplicación, configurar sus opciones y cada vez que comenzamos una operación que usa el estado de OpenGL, vinculamos el objeto con nuestra configuración preferida. Hay objetos, por ejemplo, que actúan como objetos contenedores para los datos del modelo 3D (una casa o un personaje) y cada vez que queremos dibujar uno de ellos, vinculamos el objeto que contiene los datos del modelo que queremos dibujar (primero creamos y configuramos opciones para estos objetos). Tener varios objetos nos permite especificar muchos modelos y siempre que queramos dibujar un modelo específico, simplemente vinculamos el objeto correspondiente antes de dibujar sin volver a configurar todas sus opciones.

2. Parámetros Básicos del Entorno

2.1. Uso de la Ventana

Lo primero que debimos configurar antes de comenzar a crear el entorno gráfico prehistórico fue crear un contexto OpenGL y una ventana de aplicación para dibujar. Sin embargo, esas operaciones son específicas por sistema operativo y OpenGL intenta abstraerse de estas operaciones a propósito. Esto significa que tenemos que crear una ventana, definir un contexto y manejar la entrada del usuario por nosotros mismos. Afortunadamente, existen bastantes bibliotecas que brindan la funcionalidad que buscábamos, algunas dirigidas específicamente a OpenGL. Utilizamos GLFW.

GLFW es una biblioteca, escrita en C, dirigida específicamente a OpenGL. GLFW nos brinda las necesidades básicas requeridas para mostrar cosas en la pantalla. Nos permite crear un contexto OpenGL, definir parámetros de ventana y manejar la entrada del usuario, que es suficiente para nuestros propósitos.

2.2. Vinculación

Para que el proyecto use GLFW, necesitábamos vincular la biblioteca con nuestro proyecto. Esto se pudo lograr especificando que queremos usar glfw3.lib en la configuración del enlazador, pero nuestro proyecto aún necesitaba saber dónde encontrar glfw3.lib ya que almacenamos nuestras bibliotecas de terceros en un directorio diferente. Por lo tanto, primero debemos agregar este directorio al proyecto.

Podemos, además, decirle al IDE que tenga en cuenta este directorio cuando necesite buscar una biblioteca e incluir archivos.

Nuestra lista de vinculaciones contempla:

$(SolutionDir)/External Libraries/GLEW/lib/Release/Win32
$(SolutionDir)/External Libraries/GLFW/lib-vc2015
$(SolutionDir)/External Libraries/SOIL2/lib
$(SolutionDir)/External Libraries/assimp/lib

2.2.1 Bibliotecas Externas Incluidas.

Para que el proyecto pudiera saber en que parte buscar las herramientas necesarias, se inserto manualmente la cadena de ubicación adecuada, el IDE también buscará en esos directorios cuando busque bibliotecas y archivos de encabezado. Tan pronto como se incluyeron las carpetas como la de GLFW, se pudo

encontrar todos los archivos de encabezado para GLFW al incluir <GLFW/..>., etc. Nuestra lista de Biblioteca incluye:

$(SolutionDir)/External Libraries/GLEW/include
$(SolutionDir)/External Libraries/GLFW/include
$(SolutionDir)/External Libraries/glm
$(SolutionDir)/External Libraries/assimp/include

2.2.2 Parámetros de Entrada del Vinculador.

Una vez que establecimos las cabeceras externas, nuestra IDE VisualStudio puede encontrar todos los archivos necesarios, y finalmente podemos vincular las bibliotecas al proyecto yendo a la pestaña Vinculador y Entrada para terminar nuestra configuración.

soil2-debug.lib;assimp-vc140-mt.lib;opengl32.lib;glew32.lib;glfw3.lib;

3 Shaders

Se utilizaron shaders para ayudar a construir el entorno prehistorico de este proyecto, Los shaders son pequeños programas que descansan en la GPU. Estos programas se ejecutan para cada sección específica de la canalización de gráficos. En un sentido básico, los shaders no son más que programas que transforman entradas en salidas. Los shaders también son programas muy aislados en el sentido de que no se les permite comunicarse entre sí; la única comunicación que tienen es a través de sus entradas y salidas.

Los shaders utilizados están escritos en el lenguaje GLSL similar a C. GLSL está diseñado para su uso con gráficos y contiene características útiles específicamente dirigidas a la manipulación de vectores y matrices.

Los shaders siempre comienzan con una declaración de versión, seguida de una lista de variables de entrada y salida, uniformes y su función principal. El punto de entrada de cada shader está en su función principal donde procesamos cualquier variable de entrada y mostramos los resultados en sus variables de salida.

Se utilizaron shaders para la carga de modelos, para la iluminación, para la ambientación (cubemaps), y para las animaciones del proyecto jurásico.

Aparecen listados en una carpeta especifica llamada Shaders y su programación es autóctona de su función.

4. Carga de Modelos

4.1 ASSIMP

Para nuestro entorno jurásico no podemos definir manualmente todos los vértices, las normales y las coordenadas de textura de formas complicadas. En cambio, lo que queremos es importar estos modelos a la aplicación; modelos cuidadosamente diseñados por nosotros en herramientas graficas como 3DS Max o Maya. Estas herramientas de modelado 3D nos permiten crear formas complicadas y aplicarles texturas a través de un proceso llamado mapeo uv. Luego, las herramientas generan automáticamente todas las coordenadas de vértice, las normales de vértice y las coordenadas de textura mientras las exportan a un formato de archivo de modelo que podemos usar. De esta forma, contamos con un extenso conjunto de herramientas para crear modelos de alta calidad sin tener que preocuparnos demasiado por los detalles técnicos. Todos los aspectos técnicos están ocultos en el archivo del modelo exportado. Sin embargo, nosotros, como programadores de gráficos, tenemos que preocuparnos por estos detalles técnicos.

Una biblioteca de importación de modelos muy popular que fue implementada para este proyecto, es Assimp, que significa Biblioteca abierta de importación de activos. Assimp puede importar docenas de formatos de archivo de modelo diferentes (y exportar a algunos también) cargando todos los datos del modelo en las estructuras de datos generalizadas de Assimp. Tan pronto como Assimp haya cargado el modelo, podemos recuperar todos los datos que necesitamos de las estructuras de datos de Assimp. Debido a que la estructura de datos de Assimp permanece igual, independientemente del tipo de formato de archivo que importamos, nos abstrae de todos los diferentes formatos de archivo que existen.

Al importar un modelo a través de Assimp, carga todo el modelo en un objeto de escena que contiene todos los datos del modelo/escena importados. Assimp luego tiene una colección de nodos donde cada nodo contiene índices de datos almacenados en el objeto de escena donde cada nodo puede tener cualquier número de hijos.

4.2 Mesh

Con Assimp pudimos cargar muchos modelos diferentes en la aplicación, pero una vez cargados, todos se almacenan en las estructuras de datos de Assimp. Lo que eventualmente necesitamos es transformar esos datos a un formato que OpenGL entienda para que podamos representar los objetos.

Una malla debería necesitar al menos un conjunto de vértices, donde cada vértice contiene un vector de posición, un vector normal y un vector de coordenadas de textura. Una malla también debe contener índices para el dibujo indexado y datos de materiales en forma de texturas (mapas difusos/especulares)

Gracias al constructor, obtuvimos grandes listas de datos de malla que podemos usar para renderizar. Necesitamos configurar los búferes apropiados y especificar el diseño del shader de vértices a través de punteros de atributo de vértice para que finalmente con la última función que necesitamos definir para que la clase Mesh esté completa es su función 'Draw'. Antes de renderizar la malla, primero queremos enlazar las texturas apropiadas antes de llamar a glDrawElements. Sin embargo, esto es algo difícil ya que no sabemos desde el principio cuántas texturas (si las hay) tiene la malla y qué tipo pueden tener.

4.3 Model

A partir de este punto en el proyecto, comenzamos a crear el código de traducción y carga real con Assimp. El objetivo es crear otra clase que represente un modelo en su totalidad, es decir, un modelo que contenga múltiples mallas, posiblemente con múltiples texturas. Un dinosaurio, por ejemplo, que mantiene garras, dientes, ojos y piel aún podría cargarse como un solo modelo. Cargaremos el modelo a través de Assimp y lo traduciremos a varios objetos de malla que hemos creado.

Teniendo en cuenta que asumimos que las rutas de los archivos de textura en los archivos del modelo son locales para el objeto del modelo real, Ejemplo. en el mismo directorio que la ubicación del propio modelo. Entonces podemos simplemente concatenar la cadena de ubicación de textura y la cadena de directorio que recuperamos anteriormente (en la función loadModel) para obtener la ruta de textura completa (es por eso que la función GetTexture también necesita la cadena de directorio).

Algunos modelos que se obtuvieron en Internet para este proyecto usaron rutas absolutas para sus ubicaciones de textura, lo que no funcionó al momento de llegar a este punto. En ese caso, editamos manualmente el archivo para usar rutas locales para las texturas (si es posible). O se reemplazaron las texturas y se inició un proceso de adaptación al modelo que se describe a continuación.

2.1 MANUAL TÉCNICO:

CRONOGRAMA Y COSTOS

CRONOGRAMA DE ACTIVIDADES REALIZADAS

ACTIVIDAD	DURACIÓN	S1	S2	S3	S4	S5	S6	S7
PLANEACIÓN	***2 hrs***							
Análisis de Requisitos	*1 hr.*							
Gestión Alcances, Elementos Utilizados	*1 hr.*							
DISEÑO	*4 hrs.*							
Descripción del Diseño	*1 hr.*							
Recopilación de elementos de Diseño	*2 hrs.*							
Acoplamiento de Componentes (plataformas)	*1 hr.*							
IMPLEMENTACIÓN	***40 hrs.***							
Programación de Requisitos	*6 hrs.*							
Búsqueda y buen uso de herramientas	*6 hrs.*							
Programación Modular	*14 hrs.*							
Revisión Preliminar	*7 hrs.*							
Documentación de Código	*4 hrs.*							
Filtrado y Análisis	*3 hrs.*							
PRUEBAS	***7 hrs.***							
Casos de uso para animaciones y requerimientos	*4 hrs.*							
Casos de uso Genéricos	*3 hrs.*							
DOCUMENTACIÓN	*23 hrs.*							
Documentación de Diseño	*5 hrs.*							
Documentación de Implementación	*5 hrs.*							
Documentación Entregable y Pruebas	*8 hrs.*							
Revisión y compilación de Entregables	*3 hrs.*							
Conclusiones de Proyecto	*2 hrs.*							

Encargado	Esparza Rivera Saúl Abraham
Encargado	Sánchez Manjarrez Andrew

RESUMEN DE COSTOS Y APROXIMACIONES

CATEGORIA	*S1*	*S2*	*S3*	*S4*	*S5*	*S6*	*S7*	
Sueldos	*$1500.*^{00}	*$2500.*^{00}	*$4750.*^{00}	*$3750.*^{00}	*$3750.*^{00}	*$1500.*^{00}	*$500.*^{00}	
Plan de Propuestas	*$1500.*^{00}							
Gastos de Desarrollo		*$1000.*^{00}	*$1000.*^{00}	*$1000.*^{00}	*$1000.*^{00}			
Administración de Herramientas		*$1000.*^{00}	*$1000.*^{00}	*$1000.*^{00}	*$1000.*^{00}			
Material y Mantenimiento de Equipos							*$1000.*^{00}	**COSTO TOTAL**
Desarrollo de Entregables Plan de Proyecto	*$500.*^{00}		*$500.*^{00}		*$1000.*^{00}	*$1500.*^{00}	*$500.*^{00}	$32,750.^{00}
							PRECIO:	$40,000.^{00}

2.2 MANUAL TÉCNICO:
PROCESO DE ELABORACIÓN Y ADAPTACIÓN

Con base en el cronograma de actividades y la culminación de la fase de Planeación y Diseño, se llegó a la fase contigua de Implementación donde colaborativamente se añadieron entradas, cambios, y eliminación de elementos para ajustarse a la entrega pactada en el análisis de requerimientos.

Entre la programación modular se establecieron métricas de trabajo para poder trabajar con el entorno jurásico.

Entre estos rubros se definieron procesos para añadir objetos;

1. Importación o creación del modelo.

   La mayoría de los modelos del proyecto son material intelectual propio, pero alrededor del 40% se obtuvieron de plataformas que ofertan diversos modelos gráficos con distintas licencias de uso.

   Los modelos obtenidos al ser de licencia gratuita no contenían texturas ni materiales. Para este punto se establecieron parámetros para su integración dentro del escenario.

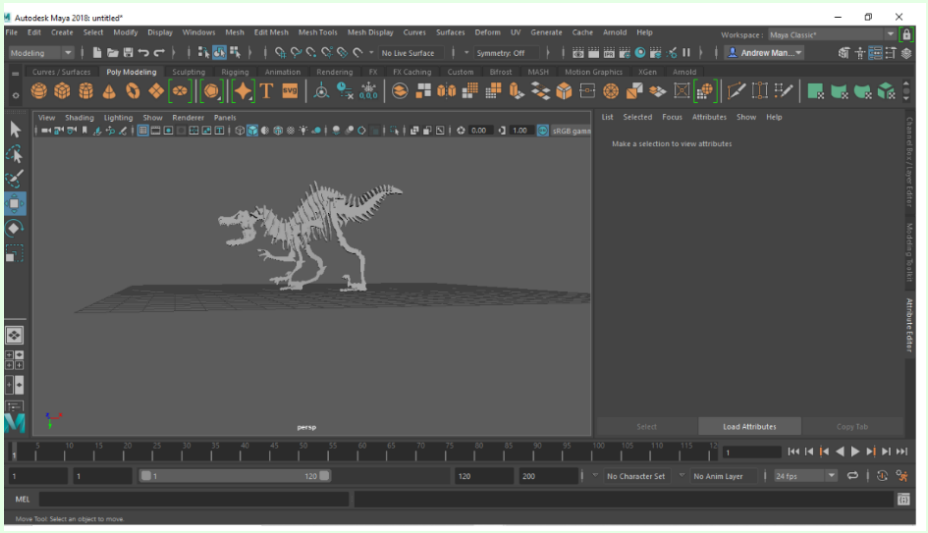

Imagen 1. Vista de interfaz de software grafico; se aprecia la importación del modelo sin texturas, y con transformaciones básicas para su acoplamiento al espacio de trabajo.

2. Selección de Texturas y adaptación de materiales.

Los modelos seleccionados y que fueron filtrados para la fase final de introducción al escenario, fueron revisitados para la toma de texturas. Se buscó la mejor calidad de materiales y se trabajaron las imágenes con el software GIMP, para poder utilizarse como textura y moldear el mapa de uv. Con la finalidad de crear una mejor ambientación al entorno jurásico.

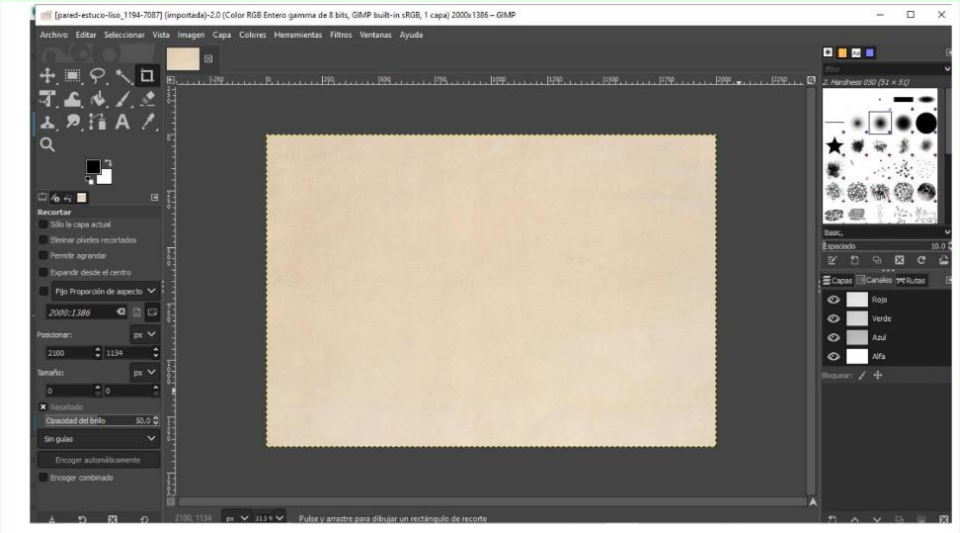

Imagen 2. Vista de interfaz de GIMP, se aprecia la importación de la imagen para la textura, se emplean herramientas propias de la plataforma para adecuar la textura, en tamaño y calidad.

3. Convergencia de materiales y finalización del modelado

Se acoplan los materiales al modelo, se vuelven a revisar transformaciones básicas del objeto para adecuar su posición dentro del escenario y finalmente, se importa para su posterior incorporación al entorno de desarrollo integrado que trabaja la integración completa del proyecto.

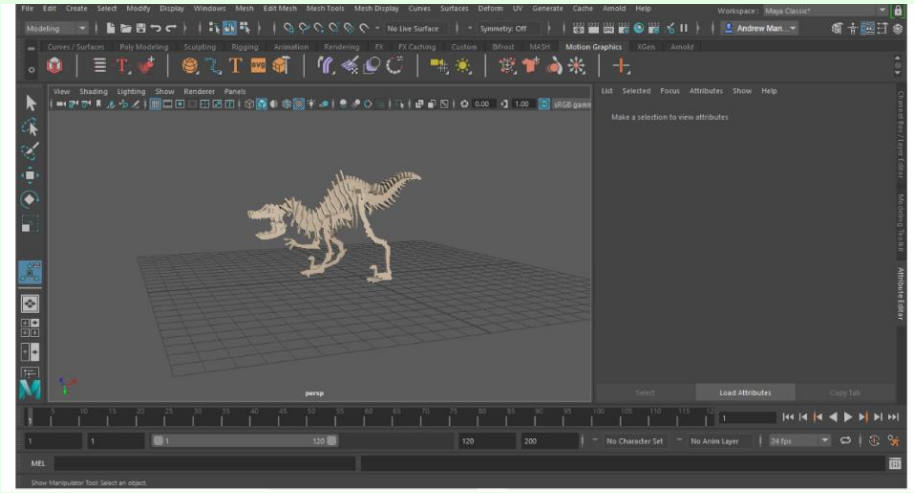

Imagen 3. Vista de interfaz de software grafico; se aprecia la forma final del modelo con texturas integradas, y con las transformaciones básicas de posicionamiento.

4. Integración del modelo al escenario final

   Finalmente, se intercambia el espacio de trabajo para continuar con la integración y acoplamiento al entregable final. Se asegura que el nuevo modelo no afecte la estabilidad del proyecto, que no existan posicionamientos no deseados, o que bien, la integración del componente sea la deseada.

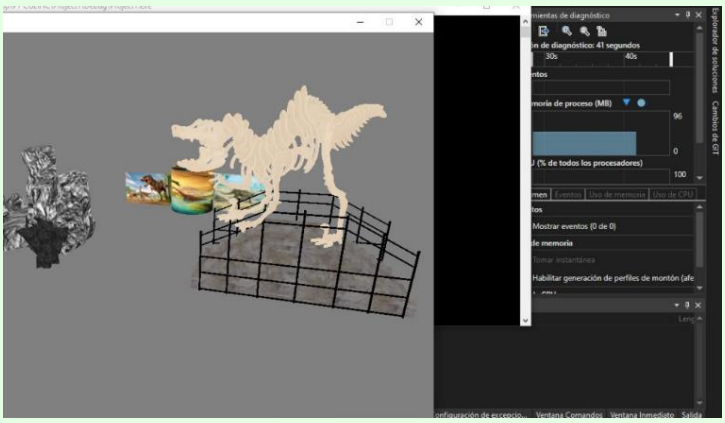

Imagen 4. Vista de interfaz de software de Visual Studio; se aprecia la forma final del modelo con los demás modelos también terminados.

5. Control de Versiones

   Se guardan los nuevos cambios y se actualizan en el repositorio central; se contemplan comentarios en la documentación como la fecha de edición, el cambio realizado y la funcionalidad.

   Se utiliza la herramienta colaborativa de GitHub para establecer un parámetro de control de versiones, respaldo y trabajo contiguo.

2.3 MANUAL TÉCNICO:
LECCIONES APRENDIDAS

Sanchez Manjarrez Andrew:

Personally, it was a vast experience to understand everyday concepts immersed not only in our language, but also in our activities; The era of animation is so present in our daily lives that sometimes it goes unnoticed, its complexity is so specific that what might seem like a simple task, could be something that can carry out months of work. However, the beauty it brings to the world makes it worth every second of effort.

The culmination of this project demonstrates a comprehensive sum of multiple pragmatic and theoretical skills acquired for the area of knowledge of computer graphics. From the background for the configuration (concepts such as shading, mesh and models) to the elaboration and adaptation process, going through the animation area where there is a mathematical process typical of engineering. To finish with the compilation of the documentation and the administrative terms of software engineering.

The objectives of this project were fully consolidated, its progress and development were according to the schedule and the established times.

The uses of this project were recreational, but computer graphics has also expanded the boundaries of art and entertainment. Movies and videogames nowdays make extensive use of computer graphics to create images that test the bounds of imagination. We see a variety of multi-uses that is only growing exponentially, the ability to quickly visualize newly designed shapes is indispensable in engineering applications and therein lies its importance.

But to me, the development of computer graphics has made possible a virtual reality, that exists only within the reach of our minds and computers.

Esparza Rivera Saul Abraham:

This is the first time that I was able to complete such a project like this one, it's the project that just keeps on giving. Since the beginning I knew it was going to be challenging but it turned out to be quite the monster task to get everything done. I remember the first lab practices; they were easy, but it felt like it was the beginning of something way bigger.

While making this project I learned a lot about computer modelling and how difficult it truly is. It gave me a better understanding of the design and programming

involved in the industry and made me realize how tuff it is to make a simulation like this one.

I speak about complexity, but it was only present during my sessions working with the good'ol Open GL, bugs, memory allocation errors, textures not showing correctly and the tedious task of running the entire project just because you forgot to move an object are a few of the problems and challenges I experienced and oh boy how I suffered with a few of them.

I also learned that there's no real low poly plants, well I mean there are a few around the internet but they make the processing super slow, especially on OpenGL, they make any simulation run 10 times slower.

I also learned that there are a lot of ways to understand your scene, for example I came up with the idea of printing on the console the exact position of the camera, and since the camera can go anywhere, it helped me a lot positioning all the elements of the scene, also while making all those objects that follow a path it was funny using the camera as a guide to get the specific coordinates before coding anything.

Another thing I learned is the importance of good modelling/texturing practices, I use 3ds Max because it's very intuitive and because for some reason I find it easy to use, learning the commands to control the view, all keyboard shortcuts, and the way to make better models is something I really appreciate because it made me feel in control, and that's a sensation we all like to experience. Overall, I think my experience with this project was good and gave me a lot of fundamentals and better practices using software that I don't really use a lot and also helped me understand the work behind all those CG productions, like videogames and movies, now I will think twice before talking bad about a movies effects, or a videogames quality.

2.4 MANUAL TÉCNICO: USO DE LICENCIAS Y SOLICITUDES

Términos y condiciones para los modelos de CGTrader

Royalty Free License
November 5, 2020
If the model is under Royalty Free license, you can use it as long as it is incorporated into the product and as long as the 3rd party cannot retrieve it on its own in both digital and physical form. You cannot resell the model you bought in its digital form and you cannot resell it in its printed form as a separate, single item.
Product may not be sold, given, or assigned to another person or entity in the form it is downloaded from the Site.
The Buyer's license to Product in this paragraph is strictly limited to Incorporated Product. Any use or republication, including sale or distribution of Product that is not Incorporated Product is strictly prohibited. For illustration, approved distribution or use of Product as Incorporated Product includes, but is not limited to:
as rendered still images or moving images; resold as part of a feature film, broadcast, or stock photography;
as purchased by a game's creators as part of a game if the Product is contained inside a proprietary format and displays inside the game during play, but not for users to re-package as goods distributed or sold inside a virtual world;
as Product published within a book, poster, t-shirt or other item;
as part of a physical object such as a toy, doll, or model.
If you use any Product in software products (such as video games, simulations, or VR-worlds) you must take all reasonable measures to prevent the end user from gaining access to the Product. Methods of safeguarding the Product include but are not limited to:
using a proprietary disc format such as Xbox 360, Playstation 3, etc.;
using a proprietary Product format;
using a proprietary and/or password protected database or resource file that stores the Product data;
encrypting the Product data.
Without prejudice the information above, the Seller grants to the Buyer who purchases license rights to Product and uses it solely as Incorporated Product a non-exclusive, worldwide, license in any medium now known or hereinafter invented to:
reproduce, post, promote, license, sell, publicly perform, publicly display, digitally perform, or transmit for promotional and commercial purposes
use any trademarks, service marks or trade names incorporated in the Product in connection with Seller material;
use the name and likeness of any individuals represented in the Product only in connection with Your material.
The resale or redistribution by the Buyer of any Product, obtained from the Site is expressly prohibited unless it is an Incorporated Product as licensed above.

We also always suggest buyers discuss the usage with the seller and gain their approval, this would make you a hundred percent sure that you can proceed with your decision of using the purchase.

Editorial License
Editorial license gives a permission to use the product only in an editorial manner, relating to events that are newsworthy, or of public interest, and may not be used for any commercial, promotional, advertising or merchandising use.
In a few instances, you may otherwise have the rights to IP in content that is under Editorial license. For instance, you may be the advertising agency for a brand/IP owner or you may be the brand/IP owner itself purchasing user generated content. Given you have the rights clearance through other means, you may use the content under Editorial License commercially. Every user failing to comply with Editorial Use restrictions takes the responsibility to prove the ownership of IP rights.
For users, who are not brand/IP owners or official affiliates, the restrictions of Editorial-licensed content usage include, but are not limited to, the following cases:
Products may not be used on any item/product created for resale such as, commercials, for-profit animations, video games, VR/AR applications, or physical products such as a merchandise or t-shirt.
Products may not be used as part of own product promotional materials, billboard, trade show or exhibit display.
The product may not be incorporated into a logo, trademark or service mark. As an example, you cannot use Editorial content to create a logo design.
Products may not be used in any insulting, abusive or otherwise unlawful manner.
The product may not be used for any commercial related purpose.

Modelos utilizados de CGTrader

1, Brigalow Acacia harpophylla 4 Free 3D Model por

KangaroOz3D

Descripción:

This 3D model was originally created with Sketchup 8 and then converted to all other 3D formats. Native format is .skp 3dsmax scene is 3ds Max 2016 version, rendered with Vray 3.00 Also known as Brigalow Spearwood or Orkor, this medium-sized tree can grow up to 25 m. and creates open woodlands known as Brigalow Belt, often in mixed community with other Acacias, Eucalypti, Casuarinas, and many underwood species. Brigalow communities generally re-sprout well after fire, except for softwood scrubs, which are more densely populated. The Brigalow Belt covers an area of 6 million ha. and has been divided into 165 different regional ecosystems, that host a large variety of often endangered native fauna. Origin : Northeast Australia (Queensland, New South Wales).

2. Tree Palm Free 3D model por

zernansuarez

Descripción:

This 3D model was originally created with Sketchup 13 and then converted to all other 3D formats. Native format is .skp 3dsmax scene is 3ds Max 2016 version, rendered with Vray 3.00 tree

3, Obelisk Egypt Free low-poly 3D model por
emelyarules

Descripción:
3d model obelisk

4. Stairs por

alberobaltic

Descripcion:

This 3D model was originally created with Sketchup 13 and then converted to all other 3D formats. Native format is .skp 3dsmax scene is 3ds Max 2016 version, rendered with Vray 3.00 Joinery company. 3D Wall panels, skirting board, stairs, door and interior design items manufacture. Company: SIA Vērtne MD Location: Latvia, Contacts: alberobaltic@gmail.com Tel. +37125664850 For more information, visit: http://albero.lv

Términos y condiciones para los modelos de TurboSquid

Royalty Free License
This is a legally binding agreement between licensee (“you”), and TurboSquid regarding your rights to use Stock Media Products from the Site under this license. “You” refers to the purchasing entity, whether that is a natural person who must be at least 18 years of age, or a corporate entity. The rights granted in this agreement are granted to the purchasing entity, its parent company, and its majority owned affiliates on a “royalty free” basis, which means that after a Purchase, there are no future royalties or payments that are required. Collectively, these rights are considered “extended uses”, and are granted to you, subject to applicable Editorial Use Restrictions described below. The license granted is wholly transferable to other parties so long it is in force and not terminated, otherwise violated, or extinguished, as set forth herein. This agreement incorporates by reference the Terms of Use as well as the Site’s policies and procedures as such.

6. Creations of Imagery.

Permitted Uses of Creations of Imagery. Subject to the following restrictions, you may use Creations of Imagery within news, film, movies, television programs, video projects, multi-media projects, theatrical display, software user interfaces; architectural renderings, Computer Games, virtual worlds, simulation and training environments; corporate communications, marketing collateral, tradeshow promotional items, booth decorations and presentations; pre-visualizations, product prototyping and research; mobile, web, print, television, and billboard advertising; online and electronic publications of blogs, literature, s ocial media, and email campaigns; website designs and layouts, desktop and mobile wallpapers, screensavers, toolbar skins; books, magazines, posters, greeting cards; apparel items, brochures, framed or printed artwork, household items, office items, lenticular prints, product packaging and manufactured products.

Restrictions on Permitted Uses of Creations of Imagery.

a. Stock Media Clearinghouse. You may NOT publish or distribute Creations of Imagery through another stock media clearinghouse, for example as part of an online marketplace for photography, clip art, video, or design templates.

b. Promotional Images. Images displayed for the promotion of Stock Media Products, such as preview images on the Stock Media Product's Product Page ("Promotional Images"), may be used in Creations of Imagery, provided that the Stock Media Product itself has been Purchased and subject to the following restrictions:

i. You may NOT use a Promotional Image that has any added element which is not included as part of the Stock Media Product. An example of this type of restricted use is if the Stock Media Product contains a 3D model of an airplane, and there is a Promotional Image of that airplane rendered over a blue sky; however, the blue sky image is not included as part of the Stock Media Product. Other prohibited examples include use of Promotional Images from movies or advertisements that may have used Stock Media Product.

ii. You may NOT use any Promotional Image that has a logo, mark, watermark, attribution, copyright or other notice superimposed on the image without prior approval from TurboSquid Support.

c. Business Logos. You may NOT use Imagery in any Creation that is a trademark, servicemark, or business logo. This restriction is included because the owners of these types of Creations typically seek exclusivity on the use of the imagery in their Creation, which is incompatible with the non-exclusive license granted to you under this agreement.

7. Creations of Computer Games and Software

Permitted Uses in Creations of Computer Games and Software. Subject to the following restrictions, you may include Stock Media Products in Creations of Computer Games, virtual worlds, simulation and training environments; mobile, desktop and web applications; and interactive electronic publications of literature such as e-books and electronic textbooks.

Restrictions on Permitted Uses of Stock Media Products in Creations of Games and Software.

a. Interactivity. Your inclusion of Stock Media Products within any such Creation is limited to uses where Stock Media Product is contained in an interactive experience for the user and not made available outside of the interactive experience. Such a permitted example of this use would be to include a 3D model of human anatomy in a medical training application, in a way that the 3D model or its environment may be manipulated or interacted with.

b. Access to Stock Media Products. You must take all reasonable and industry standard measures to prevent other parties from gaining access to Stock Media Products. Stock Media Products must be contained in proprietary formats so that they cannot be opened or

imported in a publicly available software application or framework, or extracted without reverse engineering. WebGL exports from Unity, Unreal, Lumberyard, and Stingray are permitted. Any other open format or format encrypted with decryptable open standards (such as an encrypted compression archive or other WebGL programs not listed here) are prohibited from using Stock Media Products. If your Creation uses WebGL and you are not sure if it qualifies, please contact use@turbosquid.com and describe your Creation in detail

1. Jurassic Park Main Gate 3D model

by vad3d

Descripcion:

Highly detailed and realistic model of the Jurassic Park Main Gate for Blender 2.92.0

2. Velociraptor 3D model

by Natanantuness

Descripcion: NA

2.5 MANUAL TÉCNICO:
DOCUMENTACIÓN

LearnOpenGL - OpenGL. (2017). Learn OpenGL. Retrieved May 18, 2021, from https://learnopengl.com/Getting-started/OpenGL

LearnOpenGL - Shaders. (2017). Learn OpenGL. Retrieved May 18, 2022, from https://learnopengl.com/Getting-started/Shaders

Using the TurboSquid Royalty Free License. (2021, April 13). TurboSquid Blog. Retrieved May 18, 2022, from https://blog.turbosquid.com/royalty-free-license/

Microsoft. (2022, January 12). *Introducción a*. Visual Studio. Retrieved May 18, 2022, from https://visualstudio.microsoft.com/es/vs/getting-started/

LearnOpenGL - Shaders. (2017). Learn OpenGL. Retrieved May 18, 2022, from https://learnopengl.com/Getting-started/Shaders

LearnOpenGL - Model. (2017). Learn OpenGL. Retrieved May 18, 2022, from https://learnopengl.com/Model-Loading/Model

LearnOpenGL - Assimp. (2017). Learn OpenGl. Retrieved May 18, 2022, from https://learnopengl.com/Model-Loading/Assimp