SIMULWIND: maintenance **SIMUL**ator for the sustainability of European **WIND** farms

Project nº 2017-1-DE02-KA202-004261

Manual técnico

Al usar el simulador SIMULWIND, acepta los términos de licencia y los términos de uso

El apoyo de la Comisión Europea para la producción de esta publicación no constituye una aprobación del contenido, el cual refleja únicamente las opiniones de los autores, y la Comisión no se hace responsable del uso que pueda hacerse de la información contenida en la misma.

Co-funded by the Erasmus+ Programme of the European Union

DT-00M REV 0

MANUAL SIMULWIND

INICIO RÁPIDO

Paso 1: Descargar SIMULWIND.

Paso 2: Descomprimir SIMULWIND (archivo .ZIP)

Paso 3: VER ANEXO 1

SILMULWIND: PROGRAMAS Y OTROS

1. PROGRAMAS Y VERSIONES

a) UNITY VERSIÓN: 2018.3.7.1F

b) BLENDER VERSIÓN: 2.8

Muy importante la versión de Unity ya que suelen existir incompatibilidades cuando abres un proyecto tanto en versiones anteriores como posteriores. Los plugins y librerías utilizadas vendrán incluidas en el propio master del proyecto sin necesidad de incorporar nada nuevo.

La versión blender 2.8 es la utilizada porque tiene la opción de poder preparar las texturas incrustadas de los modelos como se debe. Versiones anteriores no permiten de forma fácil incluir esta opción.

2. MODELO DEL AEROGENERADOR En blender.

En el archivo blender viene el aerogenerador completo que servirá de base para incorporar nuevos elementos en su posición correcta. (Ver explicación de exportación en Blender para que la posición, texturas del objeto sea la adecuada).

Importante: NO SE DEBE EXPORTAR TODO EL MODELO, solo lo nuevo OBJETO que se quiera incluir pero se debe tener la referencia del modelo para la escala y posición.

3. PREPARACIÓN DE NUEVOS OBJETOS 3D (BLENDER).

Podremos tener dos tipos de objetos básicamente:

Los que están en el escenario o en el aerogenerador. Para los que se debe tener en cuenta el modelo del aerogenerador en blender anteriormente mencionado. Por razones de escala y posición. Desde el administrador se puede modificar escala, rotación y tamaño de objetos pero siempre es mejor tenerlo todo preparado antes y evitarán tiempos en la edición o preparación de la práctica.

Los objetos tipo Herramientas, Epis, Materiales: Donde no es necesario tener en el modelo en cuenta la escala ya que el programa permite escalar. Pero siempre es recomendable trabajar pensando en su medida real para que vaya en su tamaño desde el principio. El centro del objeto debería estar colocado donde queramos que simule su posición en la mano. Aunque desde nuestro programa hay opción de modificar la posición y demás para colocarlo.

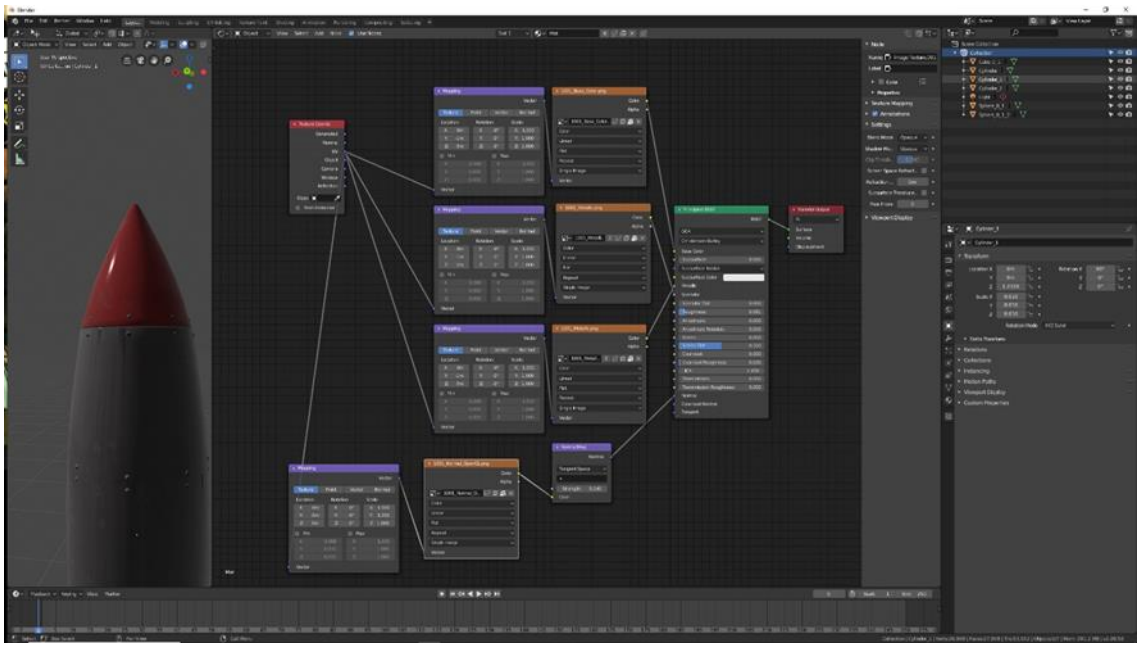
El resto de consideraciones es común a estos dos tipos de objetos:

Formatos admitidos: FBX o GLB. Se entregará un modelo en blender y FBX de ejemplo para que se vea la incrustación de texturas y parámetros de exportación para que la escala y posición sean las adecuadas.

A) Malla Low Polizada única. Conseguir el menor número de polígonos por objeto será el objetivo para que todo funcione como debe. Si intentan meter un objeto por ejemplo de 1Millon de polígonos hará suspender el programa, o hacerle ir más lento. Lo ideal sería no superar objetos nuevos de más de 10-15 mill polígonos cada uno para una máxima optimización. Ya que al tener muchas partes el escenario del aerogenerador al final hay muchísimos polígonos.

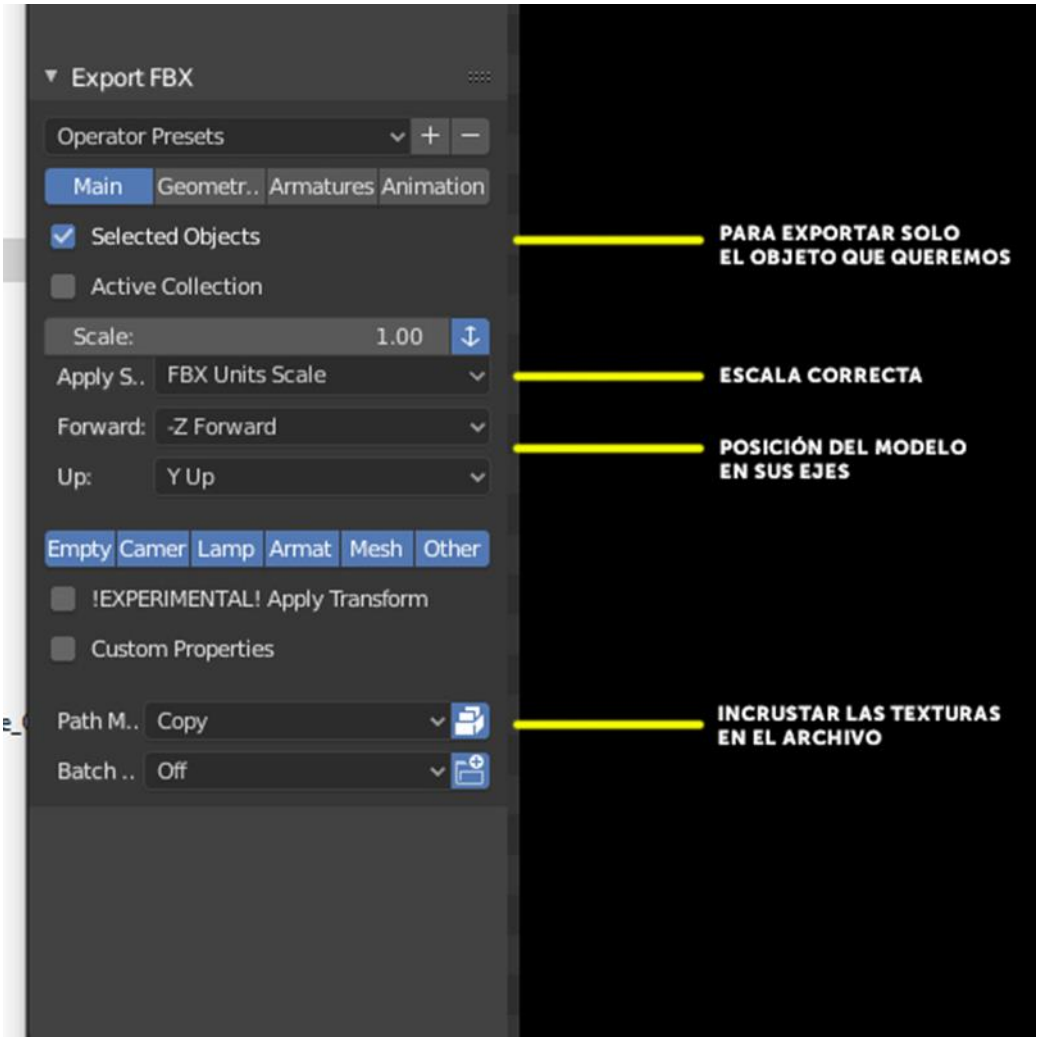
Los objetos deberán estar en una malla única sin que estén anidados unos en otros. Esto es así para que luego el área de interacción de los objetos sea la correcta.

B) Texturas incrustadas: Para conseguir un efecto óptimo las texturas se debe texturizar en PBR para que la visualización del objeto no sea plano. Cada objeto siguiendo la siguiente figura para composición del material y que luego vengan las texturas incrustadas:



Calidad de las texturas: Se recomienda una calidad no superior a 2048 pixels por textura, pero si es posible y no queda mal intentaría poner texturas de 512 pixels para que luego su carga externa sea rápida en real time en la aplicación.

C) Exportación y parámetros. Se recomienda usar en fbx aunque también se admiten otros formatos como GLB. (son los dos únicos formatos que integran la textura en el objeto).



4. OTROS:

La apliación de Simulwind tiene multitud de controles y opciones para luego colocar los objetos y editarlos para su colocación y uso dentro del aerogenerador.

Por ejemplo se puede:

Decir que el objeto sea estático o no. Si es estático significa que no se interactúa con el pero se puede ver y chocar o no con él. Si es dinámico se podrá interactuar con él en cada práctica. En cada ejercicio habrá que decirle cual es el tipo de interacción que nos interesa.

Se puede dejar objetos invisibles pero con área de colisión, esto significa que puedes usar objetos como la zapata que esta físicamente en su posición correcta pero no se ven hasta que se interactúa con ellos. O para otros usos como si quieres hacer una práctica dentro de la aspa para hacer una superficie invisible para que el player en la versión 2D no caiga dentro del aspa. Esta propiedad además se puede definir dentro de cada ejercicio por si nos interesa un caso u otro.

ESPECIFICACIÓN PLANTILLA SIMULWIND

MODELADO 3D Y SU ADAPTACIÓN LOW POLY PARA VR

Definición de modelado 3D

El modelado 3D es el proceso de desarrollo de una representación matemática de cualquier objeto tridimensional (ya sea inanimado o vivo) a través de un software especializado. Al producto se le llama modelo 3D. Se puede visualizar como una imagen bidimensional mediante un proceso llamado renderizado 3D o utilizar en una simulación por computadora de fenómenos físicos. El modelo también se puede crear físicamente usando dispositivos de impresión 3D.

Los modelos pueden ser creados automática o manualmente. El proceso manual de preparar la información geométrica para los gráficos 3D es similar al de las artes plásticas y la escultura.

El software de modelado 3D es un tipo de software de gráficos 3D utilizado para producir modelos tridimensionales. Los programas individuales de este tipo son llamados «Aplicaciones de modelado» o «modeladores». En nuestro caso el Blender.

Modelado

Los modelos 3D representan un objeto tridimensional usando una colección de puntos en el espacio dentro de un espacio 3D, conectados por varias entidades geométricas tales como triángulos, líneas, superficies curvas, etc. Siendo una colección de datos (puntos y otro tipo de información), los modelos 3D pueden ser hechos a mano, a través de algoritmos o bien escaneados.

Los modelos 3D son ampliamente usados en gráficos 3D. De hecho, su uso pre-data se extiende al uso de gráficos 3D en ordenadores. Algunos videojuegos usan imágenes pre- renderizadas de modelos 3D como sprites antes de que los ordenadores pudieran renderizarlas en tiempo real.

Hoy en día, los modelos 3D son usados en una amplia variedad de campos. La industria médica usa modelos detallados de órganos; esto puede ser creado con múltiples partes de imágenes 2-D de un MRI o escáner CT. La industria del cine lo usa como personajes y objetos para la animación o la realidad motion pictures. La industria de los videojuegos video game industry los utiliza como recurso para videojuegos. El sector científico los utiliza como modelos altamente detallados de componentes químicos. La industria de la arquitectura los utiliza para demostrar las propuestas de edificios y panoramas a través de Software Architectural Models. La comunidad ingeniera lo utiliza para diseños de nuevos artefactos, vehículos y estructuras así como portador de otros usos. En décadas recientes la comunidad de las ciencias de la tierra ha empezado a construir modelos geológicos 3D como una práctica estándar. Los modelos 3D también pueden ser la base para los aparatos físicos que son construidos con impresoras 3D o CNC machines.

Representación

Casi todos los modelos 3D pueden ser divididos en dos categorías:

Sólidos - Estos modelos definen el volumen del objeto que representan (como una roca). Estos son más realistas, pero más difíciles de construir. Los modelos sólidos son mayormente usados para simulaciones no visuales, tales como médicas y de ingeniería.

Carcasa/contorno - Estos modelos representan la superficie, ej. el contorno del objeto, no su volumen (como un cascarón infinito). Es más fácil trabajar con ellos que con modelos sólidos.

Casi todos los modelos visuales usados en juegos y películas son modelos protectores.

La apariencia de un objeto depende en gran parte del exterior del objeto, boundary representations comúnmente en gráficos de computadora. Las superficies de dos dimensiones surfaces son una buena analogía de los objetos usados en gráficos, aunque bastante a menudo estos objetos son no-manifold. Desde que las superficies no son finitas, una aproximación digital discreta es requerida: polygonal meshes (y a

una menor extensión subdivision surfaces) son por lejos la representación más común, aunque point-based las representaciones han ido ganando algo de popularidad en los años recientes. Level sets son una representación útil para deformar las superficies donde someten varios cambios tipológicos tales como fluidos.

Este modelo de carcasa es el que utilizaremos para el Low Poly y la VR

El proceso de la transformación de representación de objetos, tales como el punto medio coordinado de esfera y un punto en su circunferencia en un polígono en la representación de una esfera, es llamado tessellation. Este paso es usado en un base de polígono renderizado, donde los objetos son divididos de presentaciones abstractas ("primitivas") tales como esferas, cones etc. Son llamadas "mallas", las cuales son redes de triángulos interconectados.

Proceso de modelado

Hay 3 formas populares de representar un modelo: Modelado de curvas, Escultura digital y Modelado poligonal. Para nuestro caso práctico de Low Poly usaremos el Modelado Poligonal.

Modelado Poligonal

Son puntos en un espacio 3D, llamados vértices, están conectados para formar un polygonal mesh. La gran mayoría de los modelos 3D hoy en día están construidos como modelos de textura poligonal, porque son flexibles y porque las computadoras pueden renderizarlos muy rápido. Sin embargo, los polígonos son planos y solamente se pueden aproximar a superficies curvas usando varios polígonos.

Explicación de Low Poly

Low poly es una malla de polígonos en gráficos de computadora en 3D que tiene un número relativamente pequeño de polígonos. Las mallas de poli baja se producen en aplicaciones en tiempo real (por ejemplo, juegos) como contraste con las mallas de polietileno altas en películas animadas y efectos especiales de la misma época. El término low poly se usa tanto en sentido técnico como descriptivo; El número de

polígonos en una malla es un factor importante para optimizar el rendimiento, pero puede dar una apariencia indeseable a los gráficos resultantes.

Las mallas poligonales son uno de los principales métodos para modelar un objeto 3D para que lo muestre una computadora. Los polígonos pueden, en teoría, tener cualquier número de lados, pero generalmente se dividen en triángulos para su visualización. En general, cuanto más triángulos en una malla, más detallado es el objeto, pero más computacionalmente requiere mostrar. Para reducir los tiempos de renderizado (es decir, aumentar la velocidad de fotogramas), se debe reducir el número de triángulos en la escena, utilizando mallas de polietileno bajas.

Las técnicas de gráficos de computadora, como la asignación normal y de relieve, se han diseñado para compensar la disminución de polígonos al hacer que un objeto de polietileno bajo parezca contener más detalles de lo que lo hace. Esto se hace alterando el sombreado de los polígonos para que contengan detalles internos que no están en la malla.

Presupuesto de polígonos

Una combinación del motor del juego o el método de renderizado y la computadora que se está utilizando define el presupuesto del polígono; el número de polígonos que pueden aparecer en una escena y aún ser renderizados a una velocidad de cuadro aceptable. Por lo tanto, el uso de mallas de polietileno de baja densidad se limita principalmente a los juegos de computadora y otro software que el usuario debe manipular objetos 3D en tiempo real, ya que la capacidad de procesamiento se limita a la de una computadora personal o consola de juegos típica y la velocidad de cuadros debe ser alta. Las imágenes generadas por computadora, por ejemplo, para películas o imágenes fijas tienen un presupuesto de polígonos más alto porque la renderización no necesita realizarse en tiempo real, lo que requiere velocidades de cuadro más altas.

Además, la capacidad de procesamiento de las computadoras en estas situaciones suele ser menos limitada, ya que a menudo se utiliza una gran red de computadoras o lo que se conoce como una granja de servidores de procesamiento. Cada cuadro puede

tardar horas en crearse, a pesar de la enorme potencia de la computadora. Un ejemplo común de la diferencia que esto hace son las secuencias de video en movimiento en los juegos de computadora que, debido a que pueden ser pre-renderizados, se ven mucho más suaves que los juegos en sí que no están en la malla.

Presupuesto de polígonos para nuestra aplicación.

Para el caso concreto de nuestra aplicación al ser un trabajo para Realidad Virtual tendrá un límite bajo en el número de polígonos de los modelos. Tendremos que hacer una distinción entre **modelos estáticos y modelos dinámicos**.

Definiremos como **modelos estáticos** aquellos objetos que tienen representación 3d pero que no requieren interacción por parte del usuario. Al no tenerse que animar en tiempo real se podrá destinar mayor cantidad de polígonos a estas mallas. Se recomienda que los objetos de este apartado no sobrepasen los **150.000 polígonos**.

Definiremos como **modelos dinámicos** aquellos objetos que tienen representación 3d y son susceptibles de interacción con el usuario en tiempo real. Para estos objetos pondremos un límite de poligonado bajo de **entre 25000 y 50000 polígonos**.

Además tendremos un **límite de polígonos por escena en Occusion mode**. Es decir pondremos un límite general de cuantos polígonos tendrá que renderizar la aplicación en un frame. Lo que no quiere decir que sean todos los polígonos que hay en una escena 3d pero sí todos los que la aplicación de Realidad Virtual tendrá que renderizar por frame. Lo ideal sería marcar el límite total de polígonos renderizados por la aplciación sea de **entre 300000 y 500000 polígonos**.

Low poly como término relativo

No hay un umbral definido para que una malla sea poli baja; low poly es siempre un término relativo y depende de (entre otros factores):

El momento en que se diseñaron las mallas y para qué hardware.

El detalle requerido en la malla final.

La forma y propiedades del objeto en cuestión.

A medida que la potencia de cálculo aumenta inevitablemente, la cantidad de polígonos que se pueden usar también aumenta. Por ejemplo, Super Mario 64 se consideraría low poly hoy, pero se consideró un logro impresionante cuando se lanzó en 1996. Del mismo modo, en 2018, usar cientos de polígonos en una hoja en el fondo de una escena se consideraría high poly. pero el uso de tantos polígonos en el personaje principal sería considerado bajo poli. De esta manera, existe un relativismo entre la importancia de los objetos y su calidad gráfica. Los objetos más importantes, como los personajes que no son jugadores, suelen contener una mayor cantidad de detalles que los objetos menos importantes, que suelen ser pequeños o en el fondo, como una brizna de hierba. Este relativismo se subvierte cuando se usa low-poly como un estilo estético intencional: cada objeto comparte la simplicidad que aporta un bajo número de polígonos, lo que hace que la distinción entre características importantes y características sin importancia sea menos clara.

Coordenadas Blender y Unity

Para el caso concreto del blender y unity tendremos que tener en cuenta que los diferentes programas de modelado y los motores gráficos pueden tener sistema de coordenadas (XYZ) distinto. por lo que a la hora de exportar los objetos tendremos que tener en cuenta que estos tengan el mismo sistema de coordenadas. Normalmente esta opción se puede definir al exportar el objeto.

Es importante tener esto en cuenta ya que la representación visual entre un programa y otro podrá variar. Lo que también afecta a sus normales (descritas en el siguiente capítulo)

Mapeo de normals

En gráficos de computadora en 3D, el mapeo normal o el mapeo de relieve Dot3, es una técnica utilizada para simular la iluminación de golpes y abolladuras, una implementación del mapeo de topes. Se utiliza para agregar detalles sin usar más polígonos. Un uso común de esta técnica es mejorar en gran medida la apariencia y los

detalles de un modelo de polígono bajo al generar un mapa normal a partir de un modelo de polígono alto o un mapa de altura. Los mapas normales normalmente se almacenan como imágenes RGB regulares donde los componentes RGB corresponden a las coordenadas X, Y y Z, respectivamente, de la superficie normal.

Además en Low Poly se usa el mapeo de normals para eliminar del modelado aquellas partes del objeto que no se van a ver y para las que no queremos representación. Habrá que tener especial cuidado en el modelado para que todas las caras del objeto tengan sus normals bien definidas y no tengamos representaciones visuales del objeto extrañas.

Bump mapping

El mapeo de relieve es una técnica en gráficos de computadora para hacer que una superficie renderizada se vea más realista simulando pequeños desplazamientos de la superficie. Sin embargo, a diferencia del mapeo de desplazamiento, la geometría de la superficie no se modifica.

En cambio, solo la normal de la superficie se modifica como si la superficie hubiera sido desplazada. La normal de la superficie modificada se utiliza luego para los cálculos de iluminación (utilizando, por ejemplo, el modelo de reflexión de Phong) para dar una apariencia de detalle en lugar de una superficie lisa.

El mapeo de relieve es mucho más rápido y consume menos recursos para el mismo nivel de detalle en comparación con el mapeo de desplazamiento porque la geometría permanece sin cambios.

También hay extensiones que modifican otras características de la superficie además de aumentar la sensación de profundidad. El mapeo de paralaje es una de esas extensiones.

La principal limitación con el mapa de relieve es que perturba solo las normales de la superficie sin cambiar la superficie subyacente. Las siluetas y las sombras, por lo tanto, no se ven afectadas, lo que es especialmente notable para desplazamientos simulados

más grandes. Esta limitación se puede superar mediante técnicas que incluyen el mapeo de desplazamiento en el que se aplican protuberancias a la superficie o utilizando una superficie isosuperficial.

Materiales (Shaders)

Aunque hemos hablado del normal map y del bump map estos dos términos están íntimamente relacionado con la definición de Materiales del objeto 3D y su texturización. En Gráficos por computadora, los Materiales son una mejora del mapeo de texturas (y un requisito previo para los efectos de sombreado avanzados) que permite que los objetos en paquetes de modelado 3D y videojuegos simulen diferentes tipos de materiales en la vida real. Por lo general, se utilizan para mejorar el realismo de las mallas poligonales y otras formas de datos de modelos 3D.

Asocian propiedades adicionales tales como: parámetros avanzados de renderización (por ejemplo, especularidad, BRDF); propiedades físicas de comportamiento (tales como fricción, densidad); o el sonido se dispara junto con la información de textura para las superficies. Por ejemplo, si una textura hace que un objeto parezca madera, suena como madera (si algo lo golpea o se raspa a lo largo de una superficie), se rompe como madera e incluso flota como madera. Si fue hecho de metal, sonará como metal, abollado como metal y se hundirá como metal. Esto permite más flexibilidad a la hora de hacer objetos en juegos.

Dichos materiales también son valiosos para la generación de procedimientos, donde una descripción de alto nivel de un modelo se puede aumentar con capas de procesamiento para producir un resultado más detallado (por ejemplo, agregando meteorización o vegetación). Las

propiedades del material también pueden estar vinculadas a los canales de animación para mostrar el comportamiento dependiente del tiempo.

Un sistema de materiales permite a un artista digital o diseñador de juegos pensar en los objetos de una manera diferente. En lugar de que el objeto sea un modelo con una textura aplicada, el objeto, o parte del objeto, está hecho de un material.

Los ejemplos de materiales principales que se encuentran en un videojuego pueden ser: madera, concreto (o piedra), metal, vidrio, tierra, agua y ropa (como alfombras, cortinas o ropa de un personaje con animación esquelética).

Materiales Unity y Blender

Los materiales pueden ser muy distintos y variados dependiendo de su finalidad. Para la aplicación 3D de aerogeneradores nos basaremos en un tipo de material llamado Standard en Unity.

Dicho material usa las siguientes capas y su combinación dejarán representar visualmente la combinación de cualquier objeto (metálico, transparente, plástico, etc). Los materiales que usemos en Blender y luego importemos en unity deberán tener la siguiente estructura ideal:

- Capa Difusse.

- Capa Metalic

- Capa Normal

- Capa Height

- Capa occlusion

Cómo mínimo se recomienda que para nuestros modelos tengamos la capa Difusse, Normal y Metalic.

El Unity Standard Shader es un sombreado incorporado con un conjunto completo de características.

Se puede utilizar para representar objetos del "mundo real", como piedra, madera, vidrio, plástico y metal, y admite una amplia gama de tipos y combinaciones de sombreadores. Sus funciones se habilitan o inhabilitan simplemente usando o no los distintos espacios y parámetros de textura en el editor de materiales.

El sombreador estándar también incorpora un modelo de iluminación avanzado llamado sombreado basado físicamente. El sombreado basado en la física (PBS) simula las interacciones entre los materiales y la luz de una manera que imita la realidad. PBS solo ha sido posible recientemente en gráficos en tiempo real. Funciona de la mejor manera posible en situaciones donde la iluminación y los materiales deben existir juntos de manera intuitiva y realista.

La idea detrás de nuestro Shader Basada en la Física es crear una forma fácil de usar para lograr una apariencia coherente y plausible en diferentes condiciones de iluminación. Modela cómo se comporta la luz en la realidad, sin utilizar múltiples modelos ad hoc que pueden o no funcionar.

Para hacerlo, sigue los principios de la física, incluida la conservación de la energía (lo que significa que los objetos nunca reflejan más luz de la que reciben), los reflejos de Fresnel (todas las superficies se vuelven más reflectantes en los ángulos de pastoreo) y cómo las superficies se ocluyen (lo que se denomina Término de geometría).), entre otros.

El Standard Shader está diseñado teniendo en cuenta las superficies duras (también conocidas como "materiales arquitectónicos"), y puede trabajar con la mayoría de los materiales del mundo real como piedra, vidrio, cerámica, latón, plata o caucho. Incluso hará un trabajo decente con materiales no duros como la piel, el cabello y la tela.

Standar Shader para unity. Parámetros ampliados

El modelo 3D que importe la aplicación de Unity y realizado en Blender deberá cumplir con los siguientes parámetros:

- Albedo Color:

El parámetro Albedo controla el color base de la superficie. Algunas veces es útil especificar un solo color para el valor de Albedo, pero es mucho más común asignar un mapa de textura para el parámetro Albedo. Esto debe representar los colores de la superficie del objeto. Es importante tener en cuenta que la textura de Albedo no debe

contener ninguna iluminación, ya que la iluminación se agregará según el contexto en el que se ve el objeto.

- Metalic:

Cuando se trabaja en el flujo de trabajo metálico (a diferencia del flujo de trabajo especular), la reflectividad y la respuesta de la luz de la superficie se modifican por el nivel metálico y el nivel de suavidad.

Las reflexiones especulares se siguen generando cuando se usa este flujo de trabajo, pero surgen de forma natural dependiendo de la configuración que se establezca para los niveles de Metallic y Smoothness, en lugar de estar explícitamente definidas.

¡El modo metálico no es solo para materiales que se supone que deben ser metálicos! Este modo se conoce como metálico debido a la forma en que tiene el control sobre la superficie metálica o no metálica.

El parámetro metálico de un material determina qué tan "similar a un metal" es la superficie. Cuando una superficie es más metálica, refleja más el ambiente y su color albedo se vuelve menos visible. A todo nivel metálico, el color de la superficie está totalmente impulsado por reflejos del entorno. Cuando una superficie es menos metálica, su color de albedo es más claro y cualquier reflejo de la superficie es visible en la parte superior del color de la superficie, en lugar de ocultarlo.

De forma predeterminada, sin textura asignada, los parámetros Metallic y Smoothness se controlan mediante un control deslizante cada uno. Esto es suficiente para algunos materiales. Sin embargo, si la superficie de su modelo tiene áreas con una mezcla de tipos de superficie en la textura de albedo, puede usar un mapa de textura para controlar cómo varían los niveles de metal y suavidad en la superficie del material. Por ejemplo, si su textura contiene la ropa de un personaje, incluyendo algunas hebillas de metal y cremalleras. Usted querría que las hebillas y las cremalleras tuvieran un valor metálico más alto que el tejido de la ropa. Para lograr esto, en lugar de usar un solo

valor de control deslizante, se puede asignar un mapa de textura que contenga un píxel más claro colores en las áreas de las hebillas y las cremalleras, y los valores más oscuros de la tela.

Con una textura asignada al parámetro Metallic, los controles deslizantes Metallic y Smoothness desaparecerán. En su lugar, los niveles metálicos del material se controlan mediante los valores en el canal rojo de la textura, y los niveles de suavidad del material se controlan mediante el canal alfa de la textura. (Esto significa que los canales verde y azul se ignoran). Esto significa que tiene una textura única que puede definir áreas como ásperas o lisas, y metálicas o no metálicas, lo cual es muy útil cuando se trabajan mapas de textura que cubren muchas áreas de un modelo con diferentes requisitos, por ejemplo, un mapa de textura de un solo carácter. A menudo incluye múltiples requisitos de superficie: zapatos de cuero, ropa de tela, piel para las manos y la cara y hebillas de metal.

- Mapa normal (mapeo de relieve):

Los mapas normales son un tipo de mapa de relieve. Son un tipo especial de textura que le permite agregar detalles de la superficie como protuberancias, surcos y rasguños a un modelo que atrapa la luz como si estuvieran representados por geometría real.

Por ejemplo, es posible que desee mostrar una superficie que tenga ranuras y tornillos o remaches en toda la superficie, como el casco de un avión. Una forma de hacer esto sería modelar estos detalles como geometría, como se muestra a continuación.

Dependiendo de la situación, normalmente no es una buena idea tener detalles tan pequeños modelados como geometría "real". A la derecha puede ver los polígonos necesarios para componer el detalle de un solo tornillo. Sobre un modelo grande con muchos detalles de superficie fina, esto requeriría un gran número de polígonos para dibujar. Para evitar esto, debemos usar un mapa normal para representar el detalle de la superficie fina y una superficie poligonal de resolución más baja para la forma más grande del modelo.

Si, en cambio, representamos este detalle con un mapa de relieve, la geometría de la superficie puede ser mucho más simple, y el detalle se representa como una textura que modula la forma en que la luz se refleja en la superficie. Esto es algo que el hardware gráfico moderno puede hacer extremadamente rápido. Su superficie de metal ahora puede ser un plano plano de baja densidad de polietileno, y los tornillos, remaches, ranuras y arañazos atraparán la luz y parecerán tener profundidad debido a la textura.

En las líneas de arte del desarrollo de juegos modernos, los artistas usarán sus aplicaciones de modelado 3D para generar mapas normales basados en modelos de fuentes de muy alta resolución.

Luego, los mapas normales se asignan a una versión del modelo preparada para el juego con una resolución más baja, de modo que los detalles originales de alta resolución se representen utilizando el mapa normal.

Ampliación de información:

https://docs.unity3d.com/Manual/StandardShaderMaterialParameterNormalMap.html

El resto de capas aumentan la representación visual del material pero no son tan importantes como los 3 anteriores descritos anteriormente. Para una breve descripción de lo que aporta cada uno:

- Capa Height.

El mapeo de altura (también conocido como mapeo de paralaje) es un concepto similar al mapeo normal, sin embargo, esta técnica es más compleja y, por lo tanto, también más costosa en términos de rendimiento. Los mapas de altura generalmente se usan junto con los mapas normales, y con frecuencia se usan para dar definición adicional a las superficies donde los mapas de textura son responsables de representar grandes protuberancias y protuberancias.

- Capa Occlusion.

El mapa de oclusión se utiliza para proporcionar información sobre qué áreas del modelo deben recibir iluminación indirecta alta o baja. La iluminación indirecta proviene de la iluminación ambiental y los reflejos, por lo que las partes cóncavas empinadas de su modelo, como una grieta o un pliegue, no recibirían mucha luz indirecta de manera realista.

Los mapas de textura de oclusión normalmente se calculan mediante aplicaciones 3D directamente desde el modelo 3D utilizando el modelador o el software de terceros.

Un mapa de oclusión es una imagen en escala de grises, con áreas indicadoras en blanco que deben recibir iluminación indirecta completa y en negro que indica iluminación indirecta. A veces, esto es tan simple como un mapa de altura en escala de grises, para superficies simples (como la textura de la pared de piedra con nudos que se muestra en el ejemplo de mapa de altura anterior).

En otras ocasiones, generar la textura de oclusión correcta es un poco más complejo. Por ejemplo, si un personaje en tu escena es usar una capucha, los bordes internos de la campana deben configurarse para una iluminación indirecta muy baja, o ninguno. En estas situaciones, los artistas a menudo producirán mapas de oclusión, utilizando aplicaciones 3D para generar automáticamente un mapa de oclusión basado en el modelo.

- Capa Emission.

La emisión controla el color y la intensidad de la luz emitida desde la superficie. Cuando usas un material emisivo en tu escena, aparece como una fuente visible de luz.

Formatos de Exportación del modelo 3D

La aplicación importará objetos con los siguientes formatos: Los 3 se pueden exportar fácilmente desde Blender.

- FBX. El recomendado.

- OBJ.

- Blender

Nomenclatura de los objetos 3D.

Para que la aplicación funcione correctamente se planteará una estructura jerárquica definida según el uso que hará de ella el motor a tiempo real. En una fase más avanzada de este proyecto se realizará el estudio básico de como deberá ser esta jerarquía de objetos 3D y que se adaptará a la programación realizada.

Escena 3D.

Cada nuevo tutorial o práctica a realizar desde la aplicación de realidad virtual importará una escena 3D con dos archivos separados.

Por un lado el modelo 3D total, con todos los submodelos correctamente escalados, posicionados y con sus materiales bien definidos según los parámetros anteriormente descritos en este informe.

Por otro lado el archivo con las herramientas 3D que se usarán en la práctica de realidad virtual. Esas herramientas deberán estar escaladas al tamaño correcto y real para que puedan ser usadas por los usuarios en la práctica.

Unidades de medida Blender / Unity

Será importante tener en cuenta las unidades de medida en Unity y Blender para que el escalado y proporción de los objetos sea la adecuada.

Las unidades son algo arbitrarias en los motores 3D, incluso en Unity. Aunque a veces habrá aspectos del motor que esperan una cierta unidad a escala real.

Elegir 1 unidad a = 1 metro es una apuesta segura en Unity para la mayoría de los proyectos. Por lo que en Blender lo óptimo será contar cada unidad de medida como 1 metro.

Plantilla y funcionamiento

El funcionamiento de la plantilla y su configuración para la fácil adquisición de datos necesarios para el modelado y su posterior integración en la base funcional de la aplicación ha representado uno de los puntos que ha generado más debate en el sino interno del desarrollo del proyecto. La búsqueda de equilibrio y peso de los tres ámbitos principales: Ingeniería (Técnica y Configuración), Programación y Diseño gráfico han propiciado un consenso que reúne las bases para mejorar la usabilidad sin dejar de lado la formalidad que requiere un proyecto de estas características.

El resultado del mismo establece:

Como base de la adquisición debe existir un (1) archivo en BLENDER conteniendo todas las piezas del modelo del aerogenerador a representar, reflejando con la máxima fidelidad la representación operativa del mismo.

Se establece que será la aplicación SIMULWIND la que importe de forma automática (previa identificación del archivo anteriormente descrito) todos los elementos necesarios para modelar la realidad virtual y el 3D (en función del modo de acceso), así como genere la biblioteca correspondiente, descomponiendo los elementos de forma unitaria en archivos BLENDER, e identificándolos según la siguiente codificación:

AERO-nn-xxxxxxxx-**REV**n

Siendo AERO- parte fija, nn el número correlativo de aerogenerador importado (representado en un solo archivo BLENDER), empezando por el 00 de nuestro proyecto.

xxxxxxxx representará una secuencia numérica con la que SIMULWIND identificará de forma automática y única cada una de las piezas del aerogenerador representado en el archivo BLENDER.

REVn será la última revisión del archivo (por defecto será 0).

De este modo para que SIMULWIND pueda realizar la importación con éxito, la plantilla a utilizar deberá ser (en nuestro caso y para el primer aerogenerador, objeto de este proyecto):

Archivo en BLENDER conteniendo todas las piezas y representación completa del aerogenerador cumpliendo con las características recomendadas en este documento con la obligatoriedad de mantener el LOW POLY con el siguiente nombre de archivo:

AERO-01-00000000-REV0

SIMULWIND (una vez ejecutado en modo administrador) permitirá la importación del archivo realizando las siguientes acciones de forma automática:

- Captura del conjunto completo
- Captura de cada elemento del conjunto por separado
- Generación de n archivos (siendo n el número de piezas contenidas en el archivo principal) en BLENDER identificándolos siguiendo la numeración descrita anteriormente

Este proceso generará una estructura de carpetas y subcarpetas dentro de los archivos correspondientes a la aplicación en el sistema de la siguiente forma:

Carpeta “AERO-01”

Subcarpeta “Archivo Principal”

AERO-01-00000000-REV0

Subcarpeta “Biblioteca Elementos”

AERO-01-nnnnnnnn-REVn / y sucesivos

SIMULWIND permitirá nombrar a cada modelo de aerogenerador para una fácil búsqueda, así mismo (en estadios posteriores del proceso descritos en otros documentos), requerirá identificar mediante texto libre las piezas del modelo del archivo principal, o en su caso importar de la biblioteca, que se utilizaran para las interacciones en las actividades formativas.

ANEXO 1

RECORRIDO POR LA APLICACIÓN