

Blender 3D en la Educación

Blender 3D en la Educación

Módulo 9: Simulaciones físicas y paseos virtuales

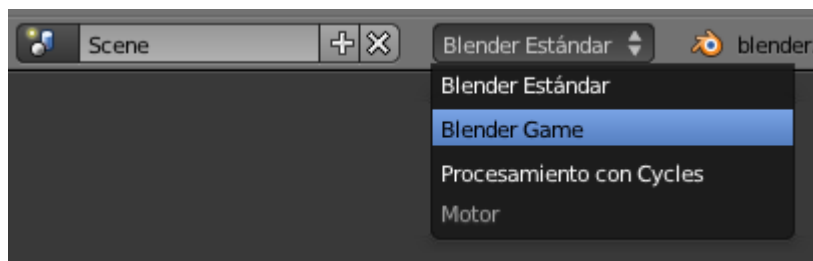
Simulaciones físicas y paseos virtuales

Ha llegado el momento de jugar, en el sentido literal del término. Blender incorpora un motor de juegos denominado **Blender Game** para simulaciones físicas, paseos virtuales o cualquier otra recreación que se nos ocurra.


En este apartado nos introducimos en la modalidad de trabajo **Blender Game** y en los fundamentos de las simulaciones físicas. Después creamos nuestro primer paseo virtual por el interior de una estancia.

El motor de juegos

Hasta ahora hemos trabajado en una modalidad denominada **Blender Estándar** donde todos los paneles, botoneras, parámetros están orientados al diseño de imágenes fijas y animaciones. Para realizar proyectos cambiamos a la modalidad **Blender Game** desde la parte alta de la interfaz cerca del menú donde cambiamos los entornos de trabajo.




Blender Game

En el paso de la modalidad **Blender Estándar** a **Blender Game** han cambiado algunas cosas aunque nos hayan pasado desapercibidas. Un simple ejemplo es el panel **Render**  donde han cambiado algunas botoneras a las que estamos acostumbrados.



El motor de juegos se ha desarrollado considerablemente desde el salto que dio Blender de la versión 2.49 a la nueva generación que comenzó con la 2.50.

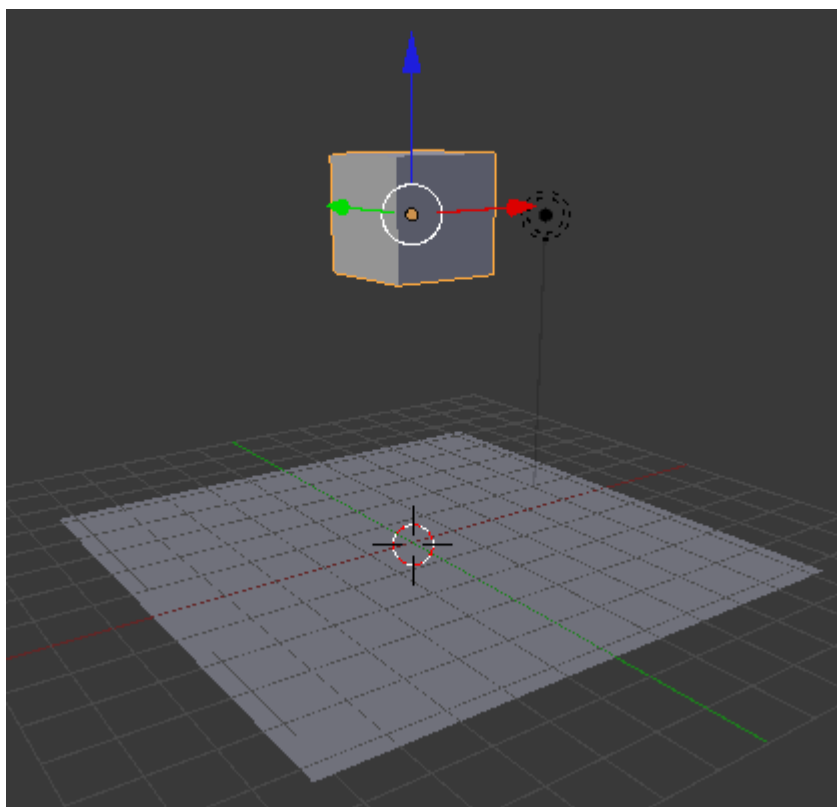
Se podrían cargar varios motores que controlaran las físicas para elegir con cuál trabajar en función del proyecto, pero de momento sólo disponemos de uno llamado **Bullet** (www.bulletphysics.org); antiguamente había otro llamado **Sumo**. Lo podemos verificar en el panel **Mundo** .





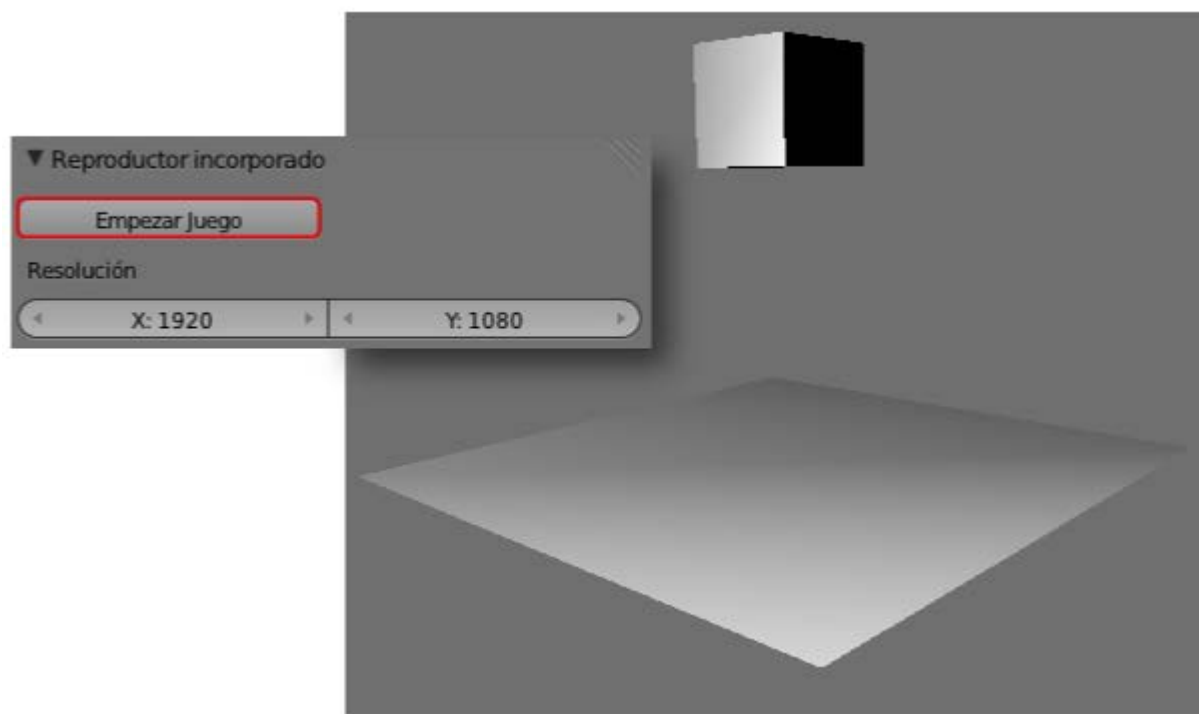
Físicas y Cuerpo rígido

No esperemos más y veamos nuestra primera simulación en acción.

A la escena inicial le sacamos un plano (**Añadir/Malla/Plano**) además de dejar el cubo por defecto. Colocamos el cubo a una distancia razonable del plano.




Y ponemos en marcha el juego; o lo que es lo mismo, activamos el motor de juego, desde el panel **Render**  pulsando el botón **Empezar juego** de la botonera **Reproductor incorporado** (no **Reproductor independiente**), aunque recomendamos el atajo "P" desde **Modo Objeto** .



Nada ocurre, salvo que el editor **Vista 3D** cambia de apariencia a una especie de *render*. Salimos del juego con la tecla "Esc" y todo vuelve a la normalidad. ¿qué ha ocurrido?. Exactamente lo que tiene que pasar: nada. Los dos objetos que forman parte de la animación (sin contar la lámpara y la cámara) son cuerpos de tipo **Estático** y por lo tanto es como si hubiéramos activado la simulación en una habitación con un suelo y una mesa; nada se mueve.

Pero estamos de acuerdo en que el cubo está en el aire y según la lógica debería caer atraído hacia el plano gracias a la fuerza de la gravedad.

Nos vamos al panel **Físicas**  y en la botonera del mismo nombre cambiamos de **Estático** a **Cuerpo rígido**.

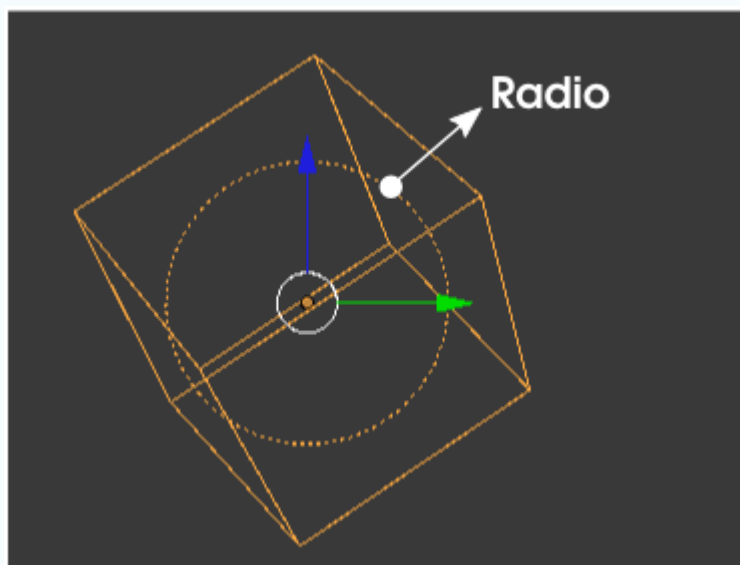


Con el puntero del ratón sobre el editor **Vista 3D** (en **Modo Objeto** , no lo olvidemos) ponemos en marcha la simulación ("P"). El cubo cae. Las físicas se ponen de manifiesto. Ya hemos hecho nuestra primera simulación.



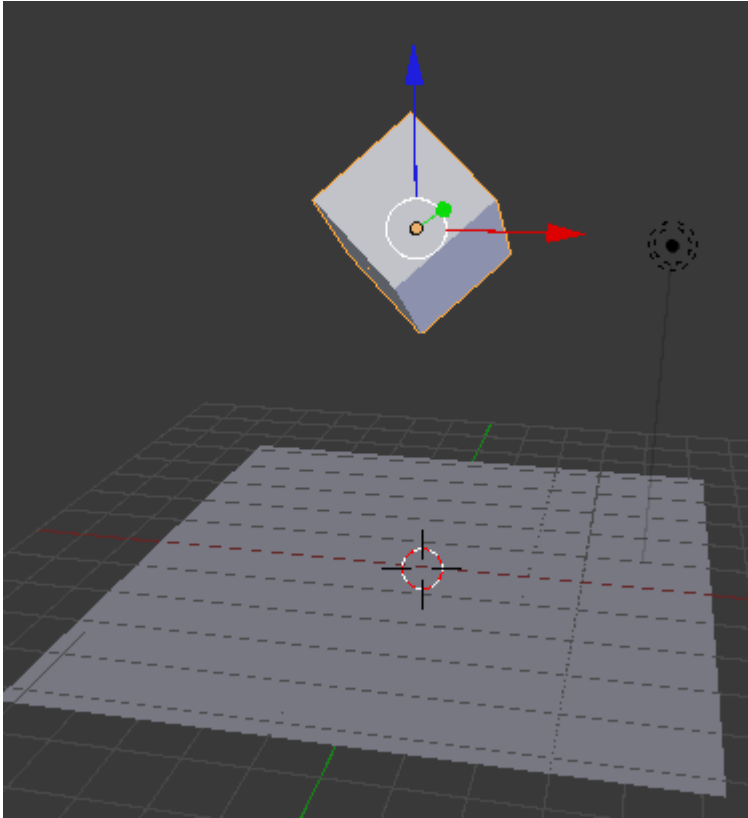
Primeras conclusiones

- Estemos en sombreado **Sólido** o en sombreado **Textura** la iluminación no es global sino que depende exclusivamente de las lámparas de la escena. El sombreado **Textura** hace visibles los mapeados UV (si los hubiera) pero algunas ayudas visuales como el **Atributo: Radio** sólo se ven con sombreado **Sólido** a **Alambre**

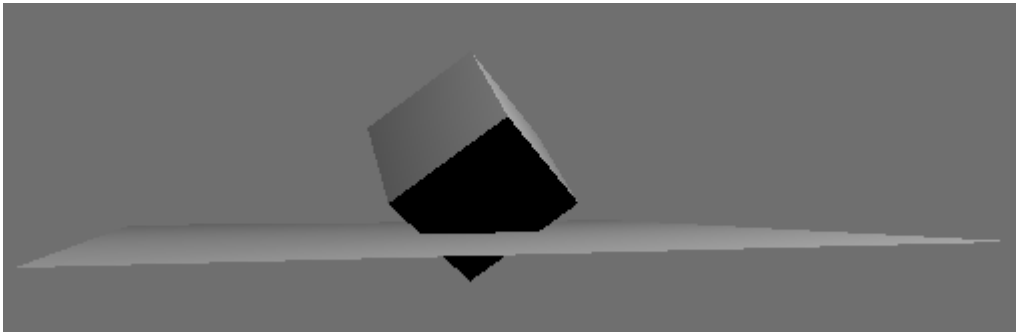



- La simulación la vemos desde el punto de vista dónde estemos trabajando, pero cuando llegue el momento la simulación debe ser encuadrada desde la cámara.
- Estas recreaciones no son cuestión de activar un par de casillas; un buen resultado depende de variar y probar un buen número de veces con distintas configuraciones.

Pero vamos a hacer algunos cambios. Lo primero es **rotar** ("R") el cubo para que no caiga con una cara paralela a la del plano del suelo.



Lo que era una simulación ("P") más o menos aceptable a pasado a ser un desastre; el cubo se queda clavado al plano (incluso lo atraviesa ligeramente) y no tiende a posarse sobre una de sus caras.



Comenzamos por activar la opción **Límites de colisión** en la botonera del mismo nombre del panel  y nos quedamos con la opción **Límites: Caja** que es perfecta para el objeto con el que estamos trabajando.



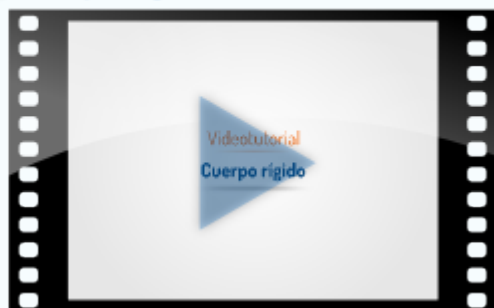
Con esto Blender determina una superficie virtual que envuelve a la malla. La simulación ("P") gana calidad: el cubo ya no atraviesa al plano y además se mueve de modo realista hasta posarse sobre una de sus caras.



Ayuda visual



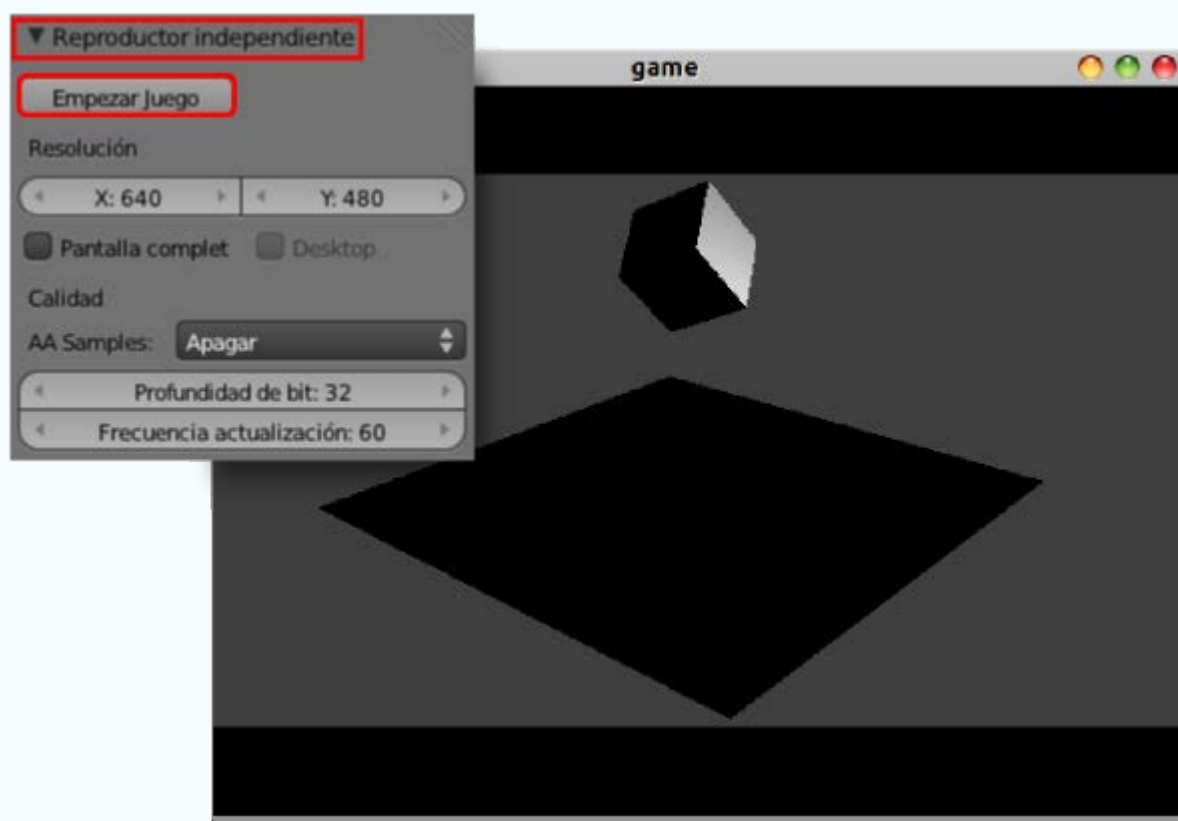
Video-tutorial Cuerpo rígido



Reproductor independiente

Lo primero que debemos saber de esta reproducción es que se toma **la cámara como punto de vista**.

Si reproducimos con **Empezar juego** de la botonera **Reproductor independiente**, Blender nos mostrará la simulación en una ventana emergente. Esta reproducción es una imitación exacta de cómo se verá cuando fabriquemos el **autoejecutable** que nos permite reproducir nuestro trabajo sin estar ejecutando Blender; incluso en un ordenador donde el programa no está ni siquiera instalado.



En nuestro ejemplo el cubo viene con un material asignado porque es el cubo que pone Blender en la escena por defecto. Pero el plano lo hemos añadido sin más y carece de material. En este **Reproductor independiente** comprobamos cómo los objetos que no tienen asignado un material aparecen completamente negros.

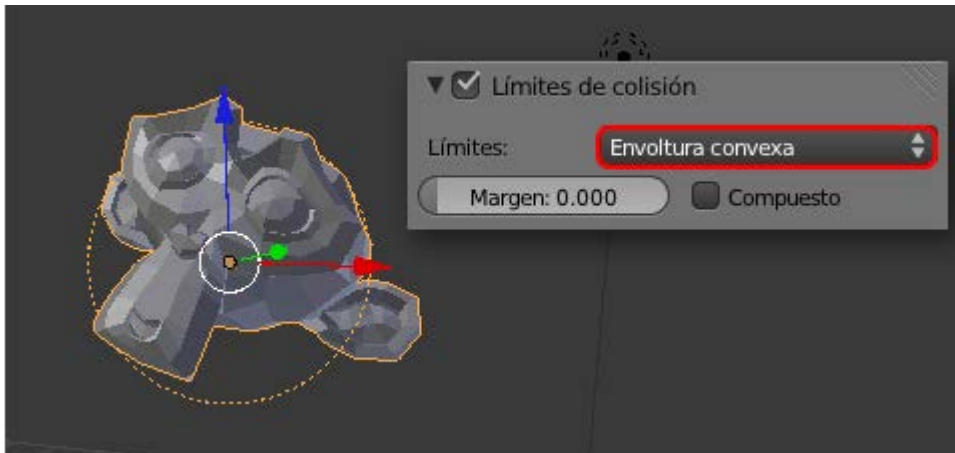
Parámetros importantes

A continuación damos un recetario de parámetros que consideraremos en primer instancia, aunque hay otros muchos que conviene

experimentar (todos son relativos a **Cuerpo rígido**):

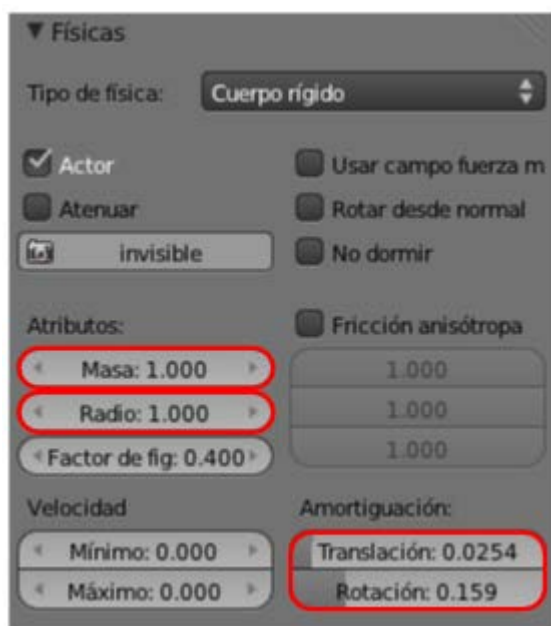
PANEL FÍSICAS . BOTONERA: LÍMITES DE COLOSIÓN

- **Límites de colisión.** Hay límites para las geometrías básicas **Cubo**, **Esfera**, **Cilindro** y **Cono**. Del resto de opciones nos interesa mucho **Envoltura convexa** que es la adecuada para mallas no tan geométricas como puede ser la mona Suzanne.



- **Margen.** Determina una distancia entre la malla y una especie de **envoltorio virtual** que es el que realmente colisiona. Si queremos que sea **0.000** y nos da algún problema basta con ampliarlo mínimamente (**0.002**, por ejemplo) y se arregla el conflicto.

PANEL FÍSICAS . BOTONERA: FÍSICAS



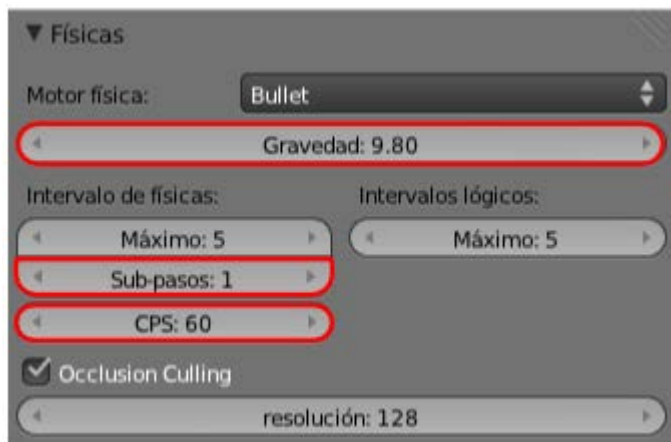
- **Radio.** Está en la botonera **Físicas**. Se representa visualmente en el editor **Vista 3D** tanto en sombreado **Alambre** como en **Sólido**. Está directamente relacionado con el **Límite de colisión: Esfera**. Si este límite está asignado a una esfera lo normal es que **Radio** coincida con la malla. Pero por diferentes motivos puede que queramos un radio mayor o menor para la esfera delimitadora.
- **Masa.** También en la botonera **Físicas**. Aunque ya sabemos que masa no es lo mismo que peso debemos hacer una interpretación en este sentido. Dos objetos, uno con una **Masa: 1.000** y otro de **2.000**, porque descenderán a la misma velocidad. Pero no es lo mismo que un cuerpo de **Masa: 10.000** golpee a otro **Cuerpo rígido** de **5.000** que a otro de **1.000**. El de **Masa: 5.000** se comportará como si fuera mucho más pesado y sufrirá menos desplazamiento que el de **1.000**.
- **Amortiguación.** En realidad no todos los cuerpos caen a la misma velocidad debido a que a algunos les afecta más que a otros el aire que se encuentran por el camino, tal es el caso de una pluma. Este rozamiento puede editarse por separado para:
-

- **Traslación**. Afecta a asuntos como la caída libre. **1.000** hace que el objeto encuentre tanta resistencia que no se mueva.



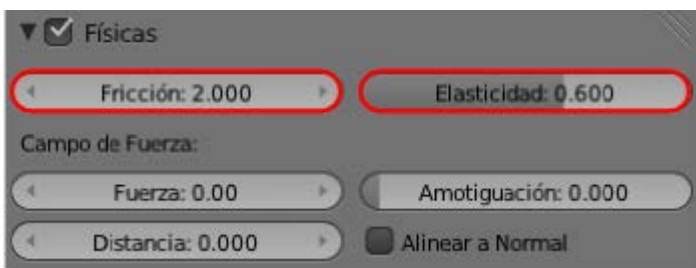
- **Rotación**. Exatamente igual.

PANEL MUNDO . BOTONERA: FÍSICAS



- **Gravedad**. Está claro que hace referencia a la fuerza con la que los cuerpos son atraídos verticalmente hacia abajo en el eje Z Global. **Gravedad: 0.00** equivale a flotar en el aire.
- **CPS**. Son los cuadros por segundo que van a calcularse. Este parámetro es realmente importante si vamos a acabar convirtiendo nuestra simulación en una animación en vídeo. En ese caso ajustaremos aquí entre 24 y 30 que serán los fotogramas por segundo a los que haremos la animación final. Un valor alto origina animaciones de tipo *slow-motión* (cámara superlenta). En realidad sea cual sea este parámetro es posible variar la velocidad de reproducción posteriormente manipulando la distancia entre los fotogramas clave que se generan al grabar.
- **Sub-pasos**. Aumenta la precisión de los cálculos corrigiendo posibles comportamientos extraños (objetos que se introducen en otros al colisionar, por ejemplo). Una variación en este parámetro puede suponer un cambio de comportamiento en los rebotes porque la información procesada es mayor.

PANEL MATERIALES . BOTONERA: FÍSICAS



Hay propiedades físicas exclusivas de los materiales. Parece razonable que cuando estamos creando un objeto con apariencia de plástico sea ahí donde nos ocupemos de determinar sus propiedades físicas para el motor de juegos. Al encontrarnos en la modalidad **Blender Game** se hace visible alguna botonera nueva como es el caso de **Físicas**.

Aquí definimos cuánta fuerza se absorbe en un impacto o si hay mucho rozamiento en el contacto de unos objetos con otros:

- **Fricción**. Determina cuánto rozamiento se origina. En un cubo que resbala por un plano inclinado, un valor de **Fricción: 0.000** en este último significa que el cubo resbala indefinidamente aunque el valor de **Fricción** de ese cubo sea enorme. El plano se comportaría como una pista de hielo.
- **Elasticidad**. En un choque con otro cuerpo ¿cuánta fuerza absorbe este objeto? Si el cubo de antes cae desde una altura, al llegar al plano sólo rebotará si ninguno de los dos (plano y cubo) absorbe el 100% de la fuerza; si uno de los dos objetos tiene **Elasticidad: 0.000** el cubo no rebotará. La correcta manipulación de este parámetro hace, por ejemplo, que una pelota bote de forma realista pero al ser un **Cuerpo rígido** no se deformará en el bote; sin embargo, el análisis de los **Cuerpos elásticos** sobrepasa los fines de **Blender: 3D en la Educación**.


De simulación a animación



¿Por qué hacer una animación?

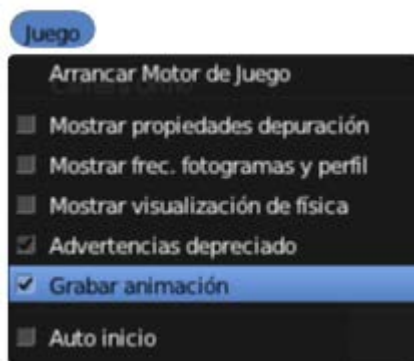
Los motivos pueden ser muy variados:

- Como no hay nada interactivo, salvo la puesta en marcha de la propia simulación, la animación se integra en web o presentaciones de una manera muy sencilla.
- En la animación se añaden materiales más sofisticados, mejores efectos de iluminación, velocidades distintas en la reproducción...
- Podemos desplazarnos a cualquiera de los fotogramas para hacer un *render* estático ("**F12**") de una de las posiciones de la simulación.

No queremos trabajar con fotogramas clave, así que es importante poner un valor de **24** en el campo **CPS** de la botonera **Físicas** del panel **Mundo** . Como hemos bajado de **60** a **24** es posible que haya que aumentar el valor de **Sub-pasos** en esa misma botonera.

Ahora generamos la animación (no el vídeo):

- Activamos la opción **Juego/Grabar animación** en la parte alta de la interfaz.




- Reproducimos la simulación ("**P**")... y cuando esté completa la paramos ("**Esc**"), como es lógico.
- Desactivamos la opción **Juego/Grabar animación**. Si no hacemos esto, cada vez que volvamos a reproducir la simulación ("**P**"), se sobrescribirá.



Ya es posible reproducir la animación ("**Alt_A**") en el editor **Vista 3D**.



Configuración del vídeo

- Pasamos a la modalidad de trabajo **Blender Estándar**.
- En el panel **Render**  ajustamos a nuestro gusto **Resolución**, **Cuadro inicial** (generalmente **1**), **Cuadro final** y atendemos a que **Velocidad de refresco** se corresponda con la que hemos definido en **CPS** anteriormente.



- En el mismo panel **Render**  vamos a la botonera **Salida** y escogemos un lugar de destino usando el icono  y el formato **MPEG**.
- Mas abajo, en la botonera **Codificación** nos aseguramos de optar por el codec **MPEG4**.



- Pulsamos el botón **Animación** y esperamos a que se genere la animación.

Material didáctico: Galileo en Pisa



En este material didáctico recordamos algunas de las posibilidades de mapeado y aprovechamos para aprender:

- A insertar una imagen y mapearla de modo automático sobre un plano.
- Usar un poco los **bloques lógicos** para aumentar la interacción con la simulación.

Y todo para ver a Galileo Galilei en lo alto de la *Torre de Pisa* poniendo en práctica su famoso experimento sobre la velocidad en la caída de los cuerpos.

Nos descargamos estas imágenes. Son PNGs con fondo transparente.



Torre de Pisa y Galileo



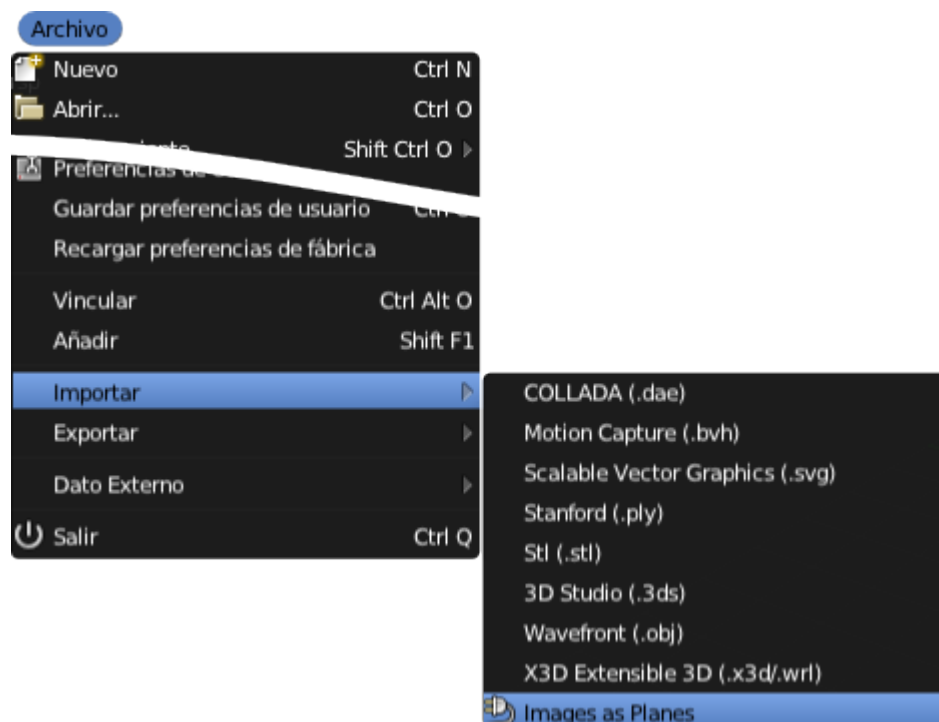
Leaning Tower of Pisa

(obra derivada) // Autor: Galileo // Autor: Joaclint
 Softeis // Licencia: CC- // Licencia: CC-SA-BY
 SA-BY-3.0 (Creative (Creative Commons)
 Commons)

Eliminamos el cubo de la escena por defecto y, de momento, no añadimos nada. Nos dirigimos a **Archivo/Preferencias de usuario** y activamos la extensión **Import-Export: Import Images as Planes**.



De regreso al entorno **Default** nos dirigimos al menú **Archivo/Importar/Imagenes al Planes** para poner en marcha esta formidable extensión.

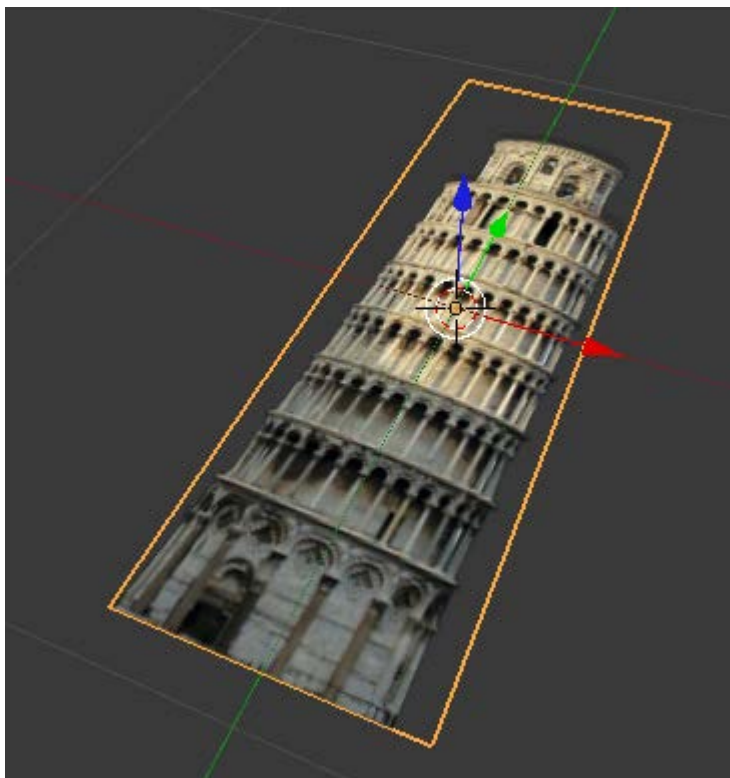


Dada la orden, toda la interfaz se convierte en un explorador de archivos con una columna de opciones a la izquierda. Antes de ir a por la imagen escogemos las opciones deseadas:



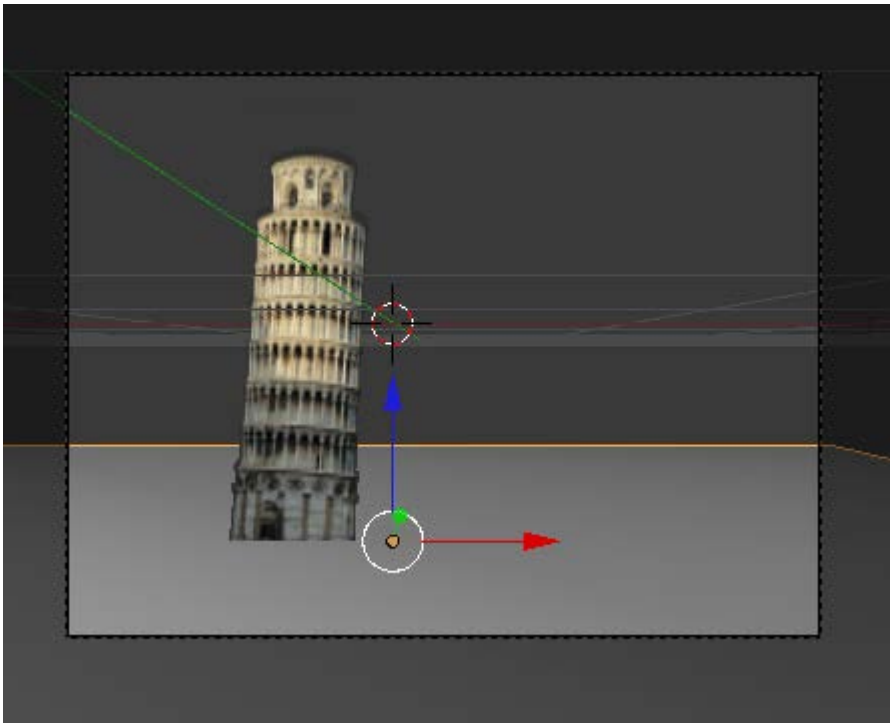
- **Sin sombra.** Importante. Al tener canal alfa, si se producen brillos en el plano se hace evidente el plano que contiene la imagen.
- **Use alpha.** Lógico ya que nuestra imagen tiene transparencia.
- **Premultiplicar.** Suaviza los bordes de las formas sobre las transparencias.
- **Use image dimensions.** Lee la información de la imagen y toma sus medidas.

Ya estamos en condiciones de ir a buscar la imagen de la *Torre de Pisa*. Este es el resultado en sombreado **Textura**.

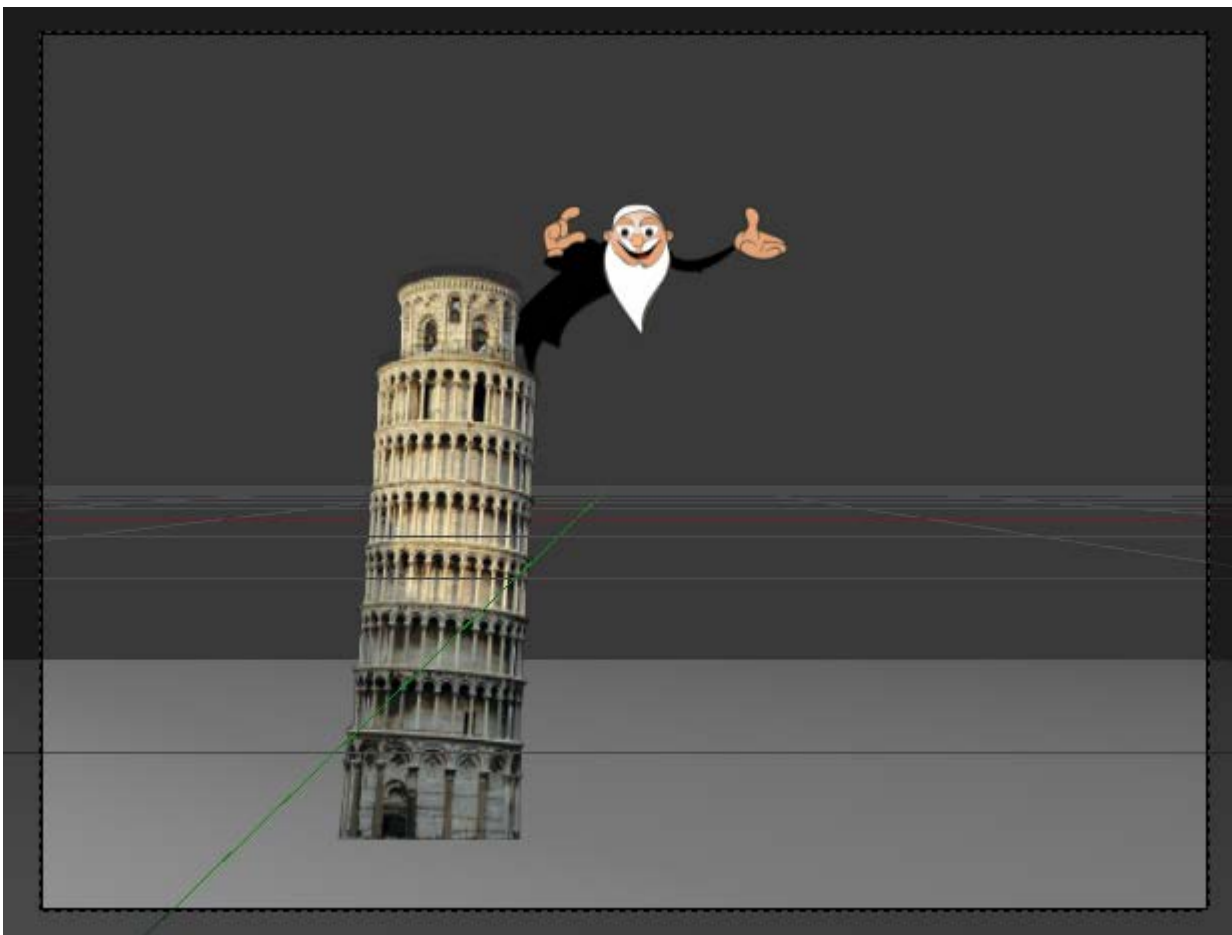


La extensión **Import Images as Planes** tiene importantes ventajas entre los que destaca que deja el material preparado tanto para la modalidad de trabajo **Blender Estandar** como para **Blender Game**.

Giramos este plano 90° en X ("RX90") y sacamos un plano (**Añadir/Malla/Plano**) para que haga de suelo.

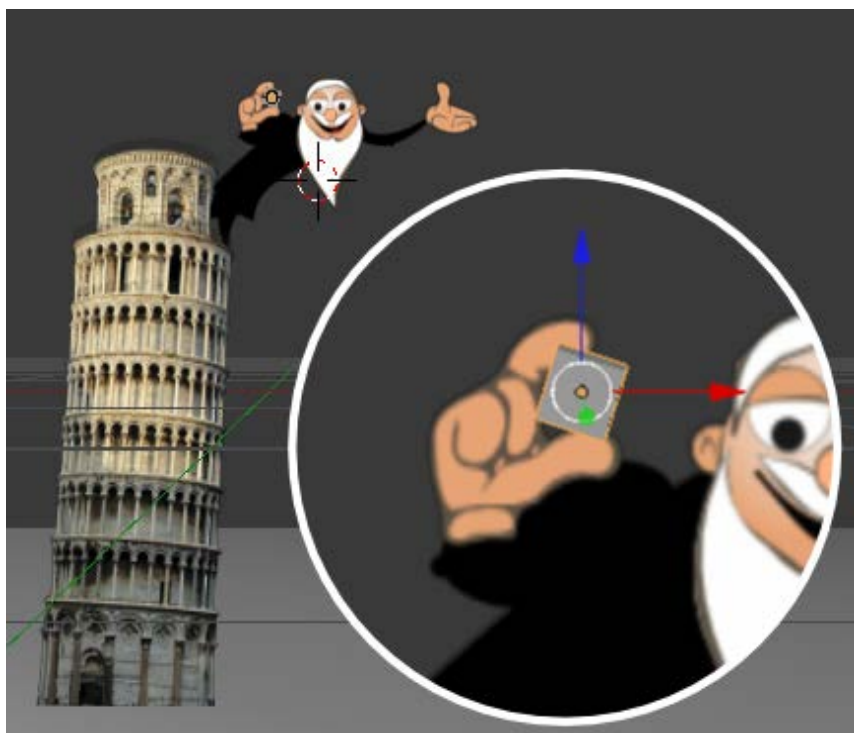


Ahora que sabemos usar la extensión **Import Images as Planes** repetimos el proceso para la imagen de Galileo. Situamos el plano algo más atrás del de la torre.

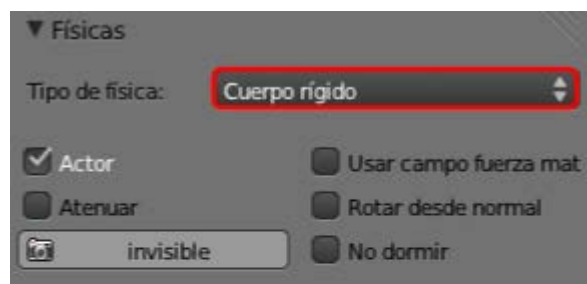


Como ya vamos a diseñar las físicas nos pasamos a la modalidad de trabajo **Blender Game**.

Sacamos un cubo (**Añadir/Malla/Cubo**, lo **escalamos ("S")**, **rotamos ("R")** y colocamos delante de la imagen de Galileo para conseguir un engaño visual como este.



En el panel **Físicas**  lo convertimos en **Cuerpo rígido** y le asignamos un **Límite de colisión: Caja**.



En el panel **Mundo** :

- Descendemos **CPS** a **24** por si queremos luego la animación.
- Para no aumentar los objetos hacemos un poco de trampa y bajamos **Gravedad** a **0.50**. Para nuestro ejemplo es una trampa más que aceptable y así el objeto tarda un tiempo razonable en recorrer toda la torre.
- Aumentamos **Sub-pasos** en el caso de que se originen problemas como que el cubo atravesase el suelo.



Tipo de física para la torre y Galileo



La mejor manera de evitar problemas con colisiones no deseadas es que tanto el plano con la torre como el de Galileo sean cuerpos **No colisiona**.



Iniciación a los bloques lógicos

No es la intención en **Blender: 3D en la Educación** aprender el uso de los bloques lógicos pero no vamos a dejar pasar la ocasión de hacer un pequeñísimo ejercicio.

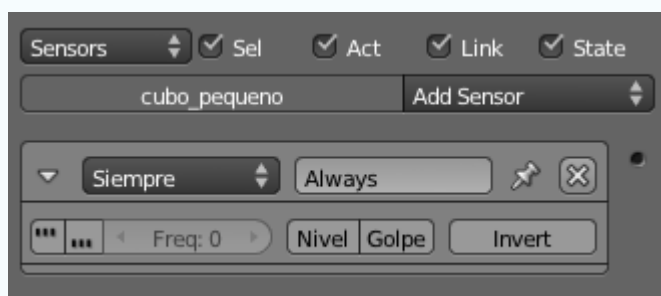
Lo que conseguimos es que la simulación no comience al ejecutar el juego sino mediante una orden nuestra a través de teclado.

Nos pasamos del entorno de trabajo **Default** al **Game Logic**. Abajo se encuentra el **Editor de lógica** . Si en el editor **Vista 3D** seleccionamos el cubo (al que hemos llamado *cubo_pequeno*) esto es lo que aparece en el **Editor de lógica** .



Tal y como hemos dicho, no entramos en detalle. Vamos directos a trabajar:

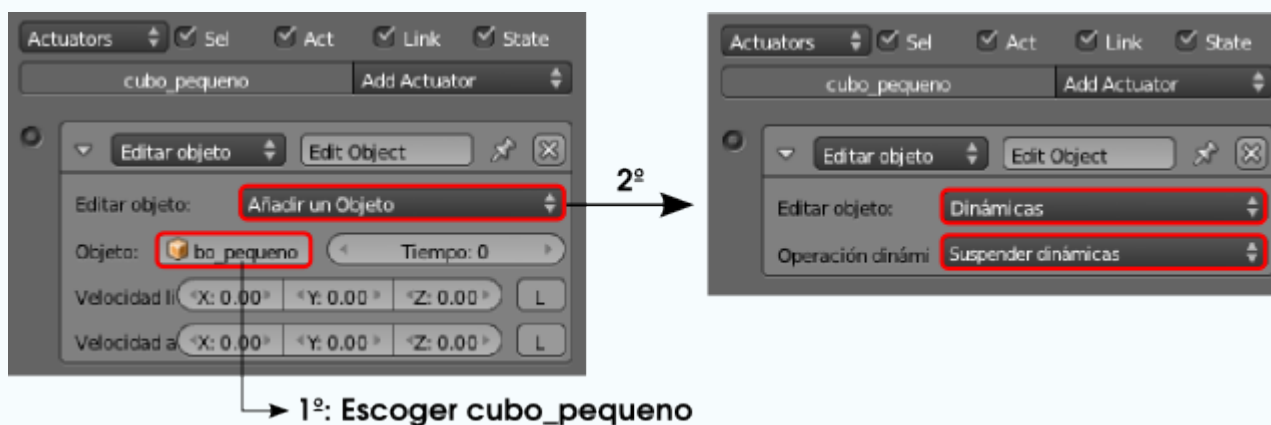
- Pulsamos sobre **Add Sensor** y escogemos uno de tipo **Siempre**.



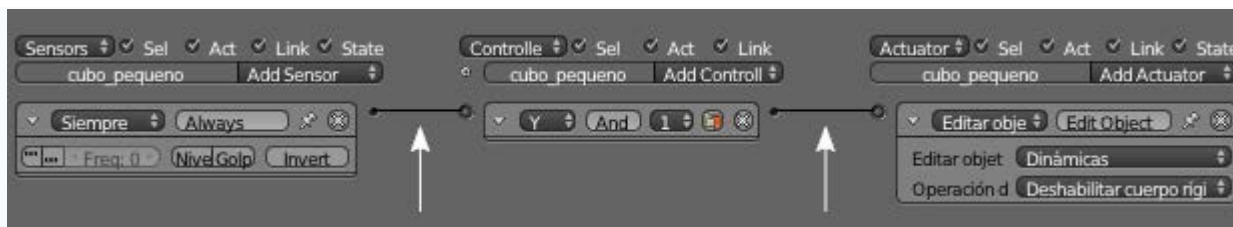
- Al lado desplegamos **Add controller** y escogemos uno de tipo **Y**.



- A la derecha pulsamos en **Add Actuator** y escogemos uno de tipo **Editar objeto**. Una vez dentro seleccionamos nuestro objeto en cuestión (*cubo_pequeno*). Sólo en ese momento pulsamos en **Añadir objeto** y seleccionamos **Dinámicas**; en el nuevo cuadro configuramos **Operación dinámica: Suspender dinámicas**.



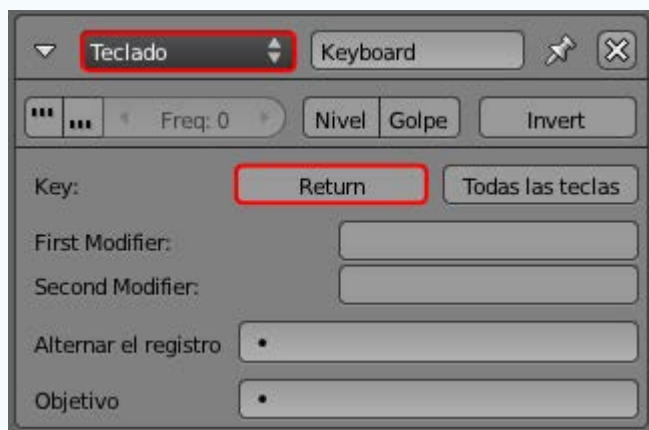
Ahora hay que relacionar los tres elementos para crear un **bloque lógico** que funcione. Sólo hay que unir con el ratón los pequeños círculos.



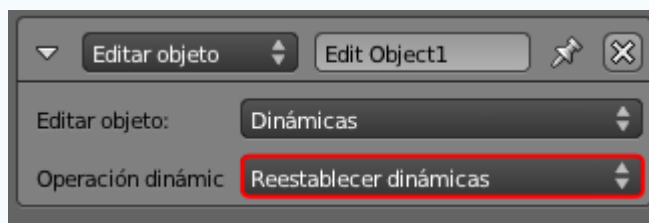
Si todo ha ido bien, al poner en marcha la simulación ("P") el cubo no caerá.

Necesitamos un segundo bloque lógico con:

- **Sensor** de tipo **Teclado**. En el cuadro que se despliega, en el campo **Key** hacemos **click** y una vez dentro pulsamos la tecla con la que queremos poner en marcha las dinámicas (nosotros optamos por "**Intro**" que Blender llama *Return*).



- **Controller** es igual que antes, de tipo **Y**. Creamos uno nuevo, no aprovechamos el anterior.
- **Actuator** también es igual solo que al final del proceso escogemos **Reestablecer dinámicas**.

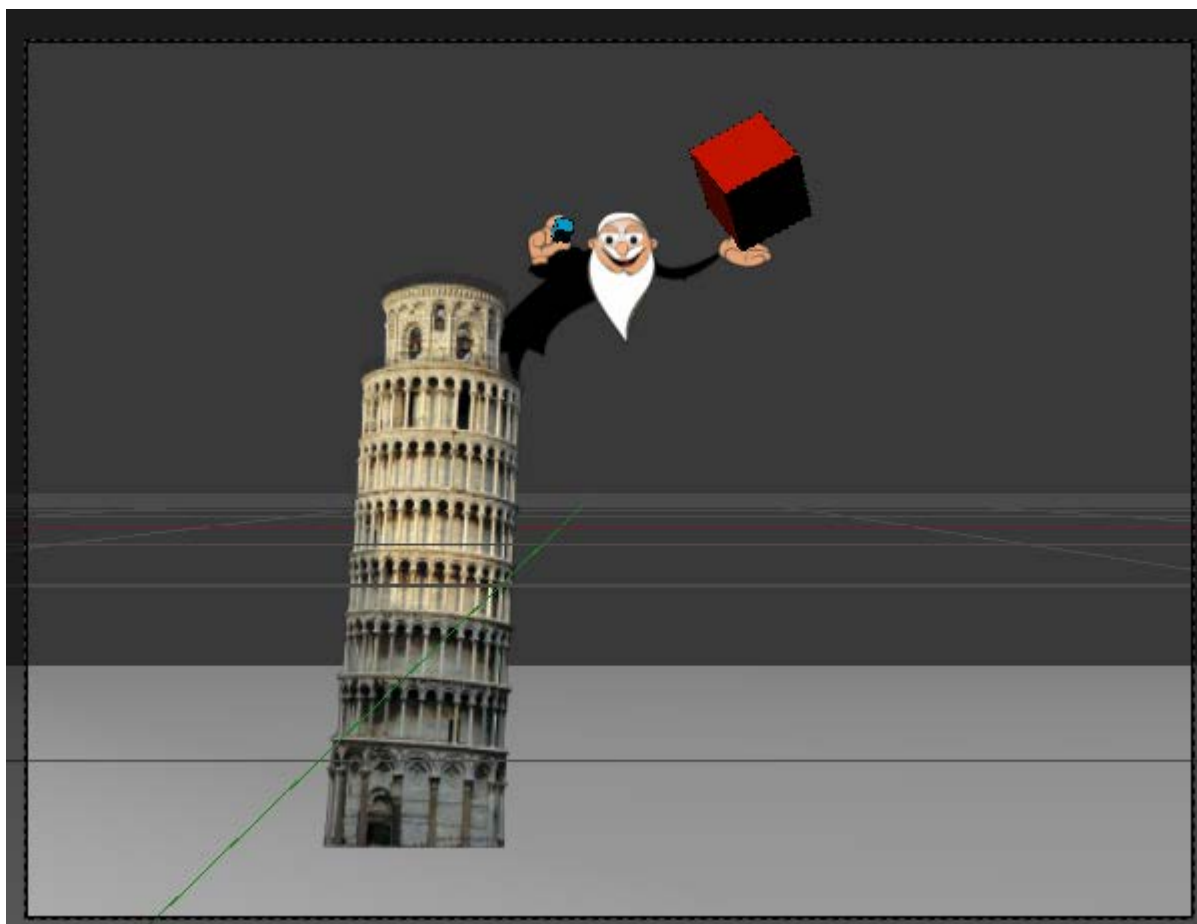


Unimos los tres elementos para que funcione el bloque lógico. Al activar la simulación ("P") el cubo sigue sin moverse, pero al pulsar la tecla "**Intro**" comienza la acción.

Hemos terminado, regresamos al entorno **Default**.

Al **duplicar** ("Shift_D") el cubo todos los datos de los bloques lógicos se heredan.

Una composición en esta línea es lo que buscamos.



Galileo está listo para comprobar si su teoría es cierta.

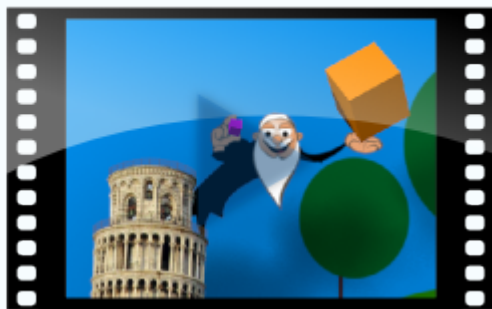
Nuestro resultado final con algun mapeado más para el fondo...



Simulación convertida en vídeo



Vídeo Galileo en Pisa



Analiza y estudia el archivo .blend

Usa este .blend para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.



Archivo
galileo.blend

Cámara interactiva

Una de las finalidades más usadas para el motor de juegos es la creación de **paseos virtuales por arquitecturas**. Las posibilidades que nos brinda este recurso son, sin duda, infinitas.

CameraFPS



Un poco de Historia

Gracias a las aportaciones de **programadores que ceden códigos** bajo licencias como Creative Commons disfrutamos de un archivo como **CameraFPS** que incluye una **cámara completamente interactiva** para recorrer un espacio arquitectónico.

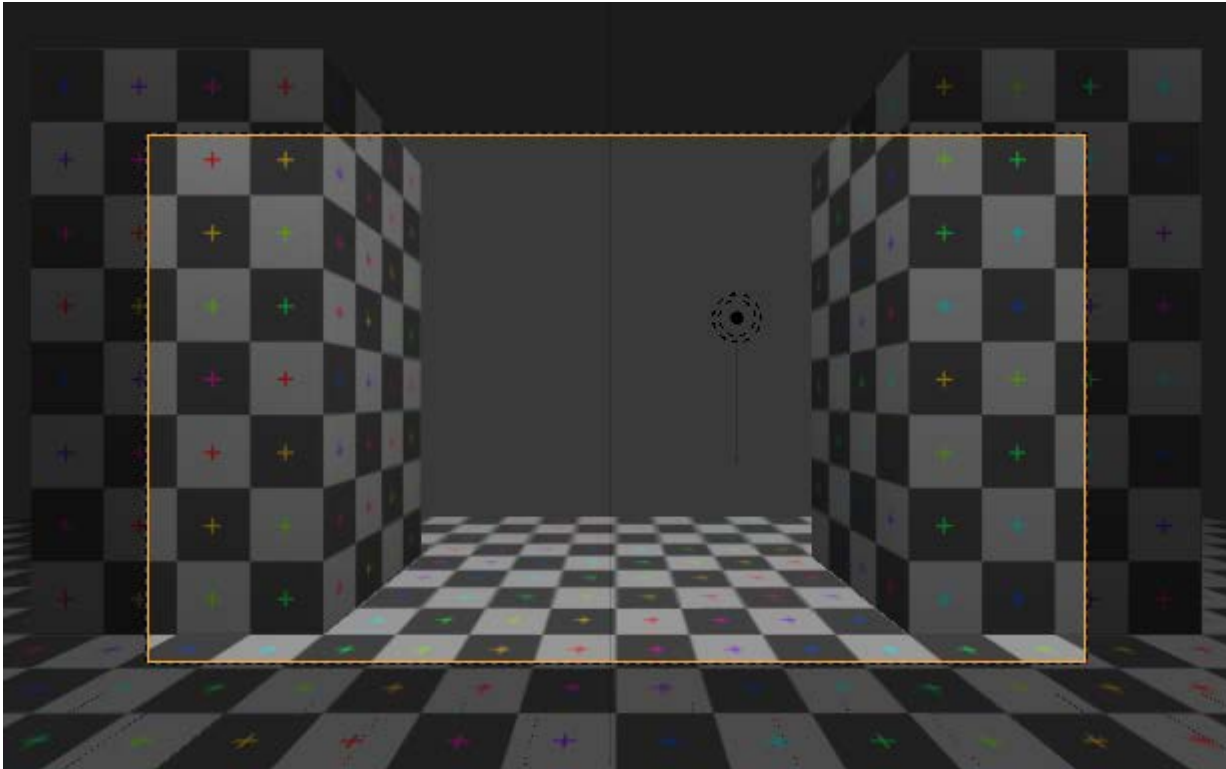
El código original **MouseLook** fue escrito por Clarck Thames y publicado con licencia CC-BY-3.0. En realidad es el resultado de un magnífico tutorial suyo en el que explica paso a paso cómo se programa el efecto. En este momento la cámara lo único que hace es obedecer el movimiento del ratón originando la sensación de cabeceo del jugador. Esta acción es en realidad la que se conoce en el mundo de los videojuegos como *mouselook* (mirar con el ratón).

El testigo lo recoge Pelle Johnsen y publica **FPSController** también con CC-BY-3.0. En esencia es una mejora de **MouseLook** al que le **añade controles para desplazar la cámara**. FPS (*First Person Shooter*) hace referencia a los videojuegos en primera persona donde la cámara representa al jugador y, por norma general, incorporan el efecto *mouselook*; esto hace que el personaje sea controlado con las dos manos:

- **Izquierda.** Para desplazamientos adelante-atrás-derecha-izquierda.
- **Ratón.** Para mirar arriba-abajo-derecha-izquierda. Si el acto de mirar se hace mientras hay un desplazamiento, entonces hay un giro del personaje.

Posteriormente Rafael Ángel López García hizo añadidos a la configuración logrando una interacción más confortable que la de **FPSController**; el resultado se llama **CameraFPS** y, por supuesto, continúa con la licencia de siempre.

CameraFPS es un *.blend* que se abre como cualquier otro, y esto es lo que muestra en el editor **Vista 3D** (desde el punto de vista de la cámara).



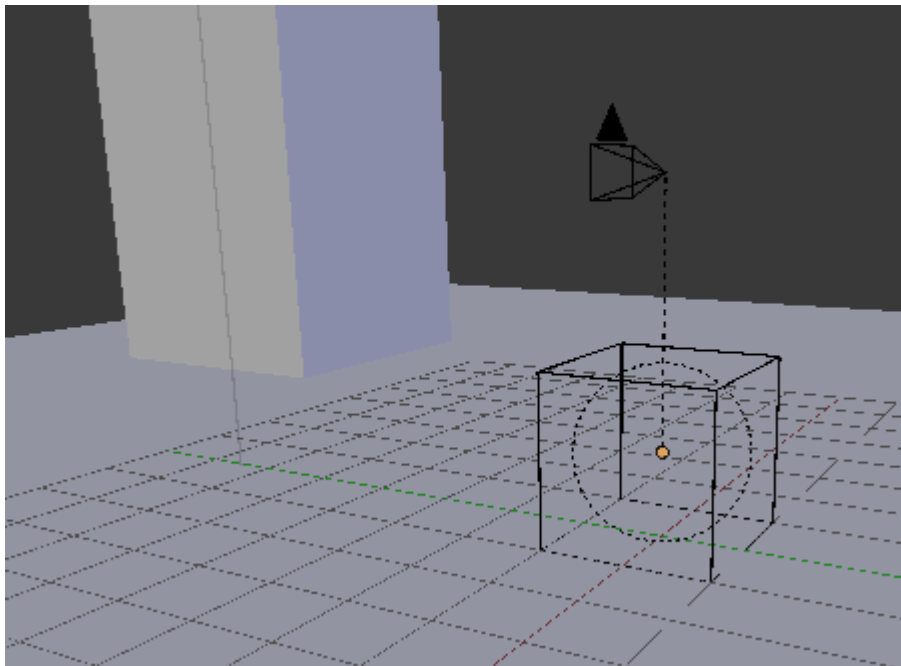
Entramos al juego ("P") y usamos los siguientes controladores:

- Tecla **"W"**. Adelante.
- Tecla **"S"**. Atrás.
- Tecla **"A"**. Izquierda.
- Tecla **"D"**. Derecha.
- Tecla **"Espacio"**. Salto.
- Movimiento del **ratón con botón izquierdo pulsado**. Mirar arriba, abajo, derecha e izquierda. Tal y como se dijo antes, si el acto de mirar se hace mientras hay un desplazamiento, entonces hay un giro de la cámara.
- Tecla **"Esc"**. Sale del juego.



No nos vamos a detener a estudiar cuál es el proceso para crear el artilugio que consigue ese comportamiento de la cámara, pero sí haremos una breve descripción.

Ya en sombreado **Sólido** y en el editor **Vista 3D**, si salimos del punto de vista de la cámara, esto es lo que hay.



- **Un suelo.** Nada sorprendente; lo único que nos puede llamar la atención es esa textura cuadriculada que tiene asignada y que se veía en sombreado **Textura**. No le damos mayor importancia porque es sólo para hacer más comprensible la experiencia de desplazarse con **CameraFPS**.
- Unas cuantas **lámparas**.
- **Un cubo.** Este objeto es el que está programado para desplazarse mientras arrastra consigo a la cámara ya que tienen una relación de parentesco.
- **Una cámara.** Asociada, como hemos dicho, al cubo. Esta cámara es la que tiene la programación para los movimientos de



Ayuda visual



Video-tutorial CameraFPS



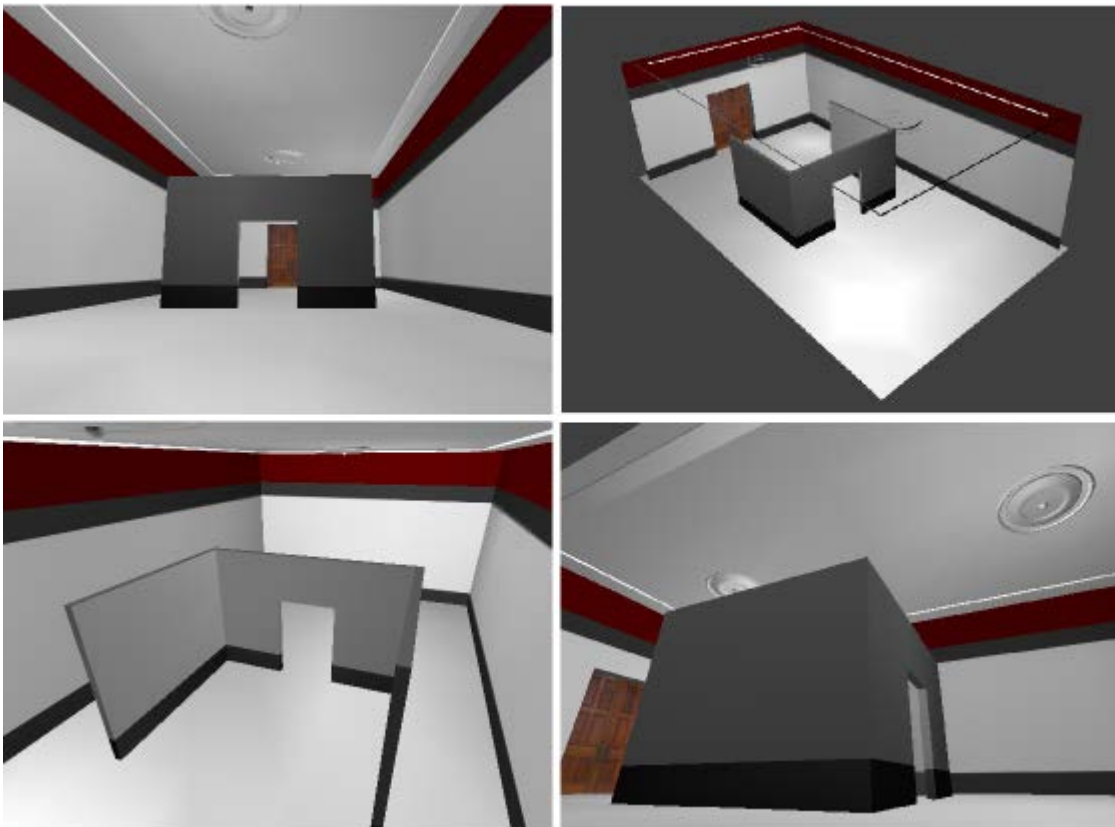
Añadir la cámara interactiva

Una vez que conocemos **CameraFPS** debemos decidir entre:

- Traer a ese *.blend* la escena que tengamos ya creada (o crearla desde cero en él)
- Llevar los dos objetos que conforman el artilugio a nuestro diseño.

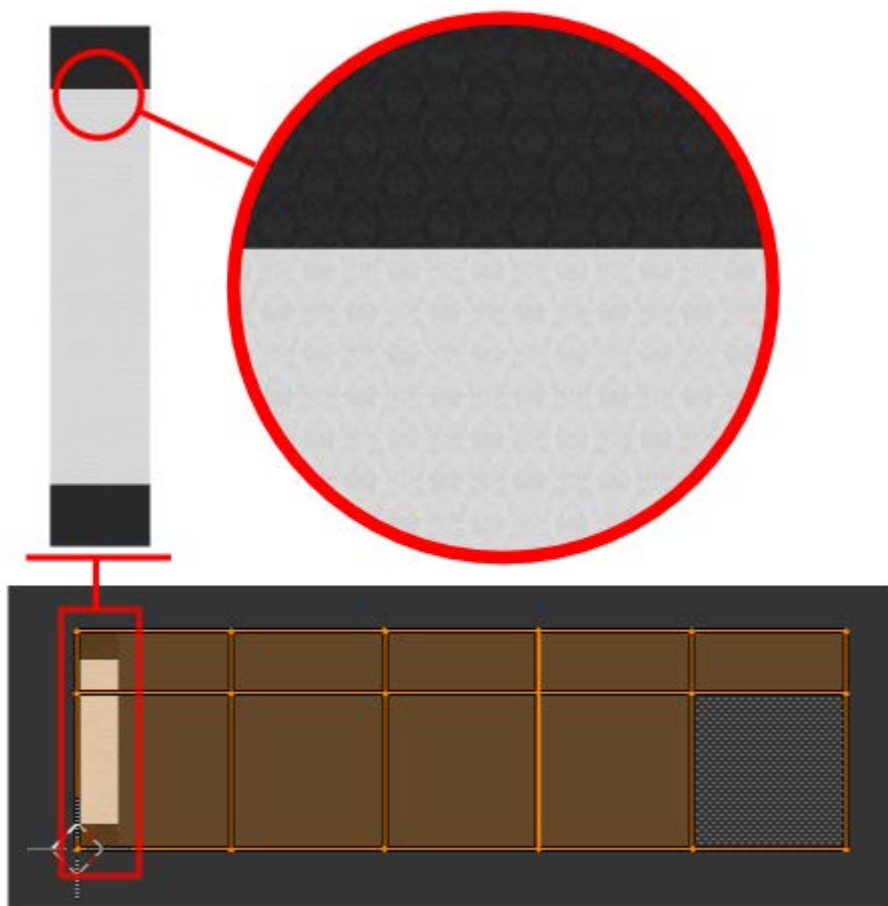
Nosotros nos quedamos con la segunda opción.

Supongamos que ya disponemos de un entorno arquitectónico por el que nos gustaría pasear como puede ser una sala de exposiciones (de momento sin muchos adornos).



Sus características:

- Se ha eliminado la cámara para evitar problemas cuando añadamos la que trae consigo **CameraFPS**.
- Tiene dos imágenes mapeadas mediante **Malla/Desplegar UVs/Desenvolver**. La de la pared es una imagen que hace las veces de módulo repitiéndose a lo largo de todas las paredes de la estancia (no del elemento arquitectónico central)



Puerta y patrón de pared

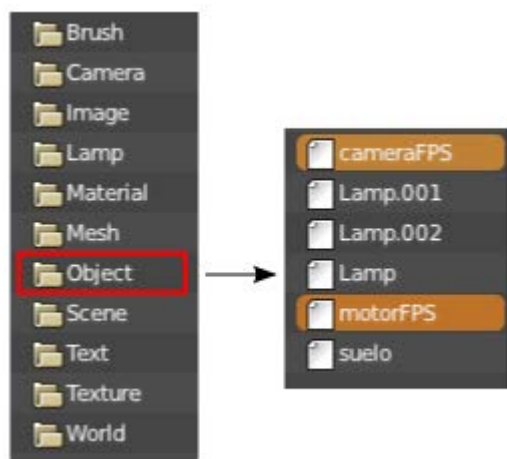


Puerta del siglo XIX
(fotografía adaptada)
Autor: Infrogmation //
Licencia: GNU Free
Documentation

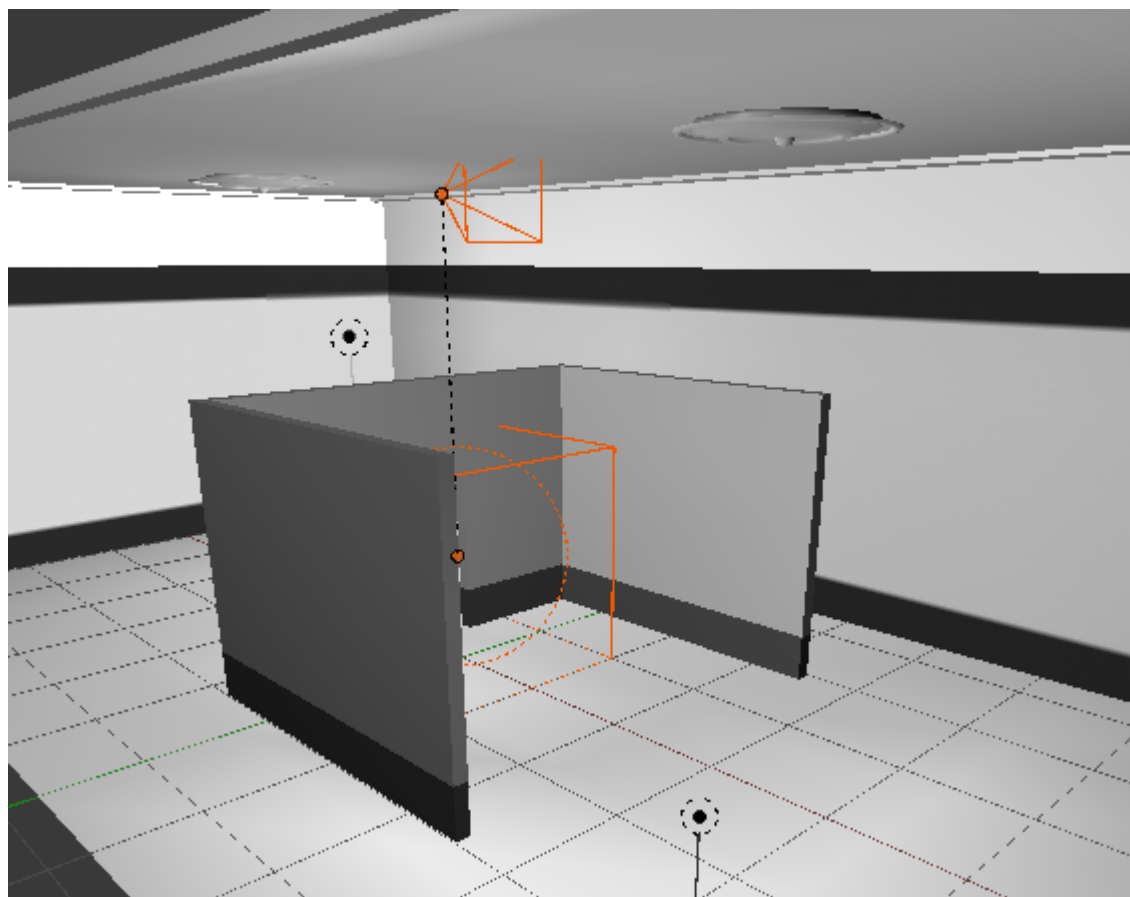


Patrón de pared
empapelada // Autor:
Joaclint // Licencia: CC-
SA-BY (Creative
Commons)

Una vez abierto nuestro entorno arquitectónico usamos el menú **Archivo/Añadir** y nos vamos a buscar el **.blend CameraFPS**; una vez dentro de sus carpetas accedemos a **Object** y del interior seleccionamos (*cameraFPS* y *motorFPS*). Después pulsamos arriba a la derecha **Link/Append from Library**.



Estos dos objetos son la cámara y el cubo propios del artilugio. Por las características del *.blend* de origen, y de las propias geometrías, el cubo aparece apoyado en la rejilla y con su **Origen** en 0,0,0.

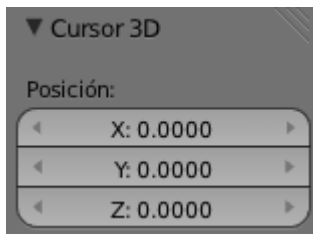


Sacamos rápido algunas consecuencias:

- El artilugio es inmenso para nuestra arquitectura.
- La precisión está garantizada porque en nuestro diseño el suelo esta con **Localización Z= 0.000**.

Pero llega el dilema: ¿escalamos la habitación o escalamos el artilugio?. El problema de no haber trabajado desde el principio con todos los datos tiene consecuencias (negativas, claro) porque cualquiera de las dos opciones es mala; y la menos mala es escalar la arquitectura. ¿Por qué? Si escalamos el artilugio no alteramos su velocidad así que el jugador parecerá que va corriendo y no andando, mientras que si escalamos la arquitectura el problema es que la iluminación cambia porque los valores de **Energía** de las lámparas no se escalan. Entre arreglar el tema de la velocidad y el de la iluminación es mucho mejor optar por lo segundo.

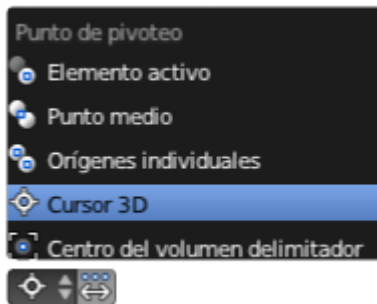
De momento nos vamos al cuadro **Propiedades ("N")** y en la botonera **Cursor 3D** hacemos que todas su coordenadas sean **0.0000**.



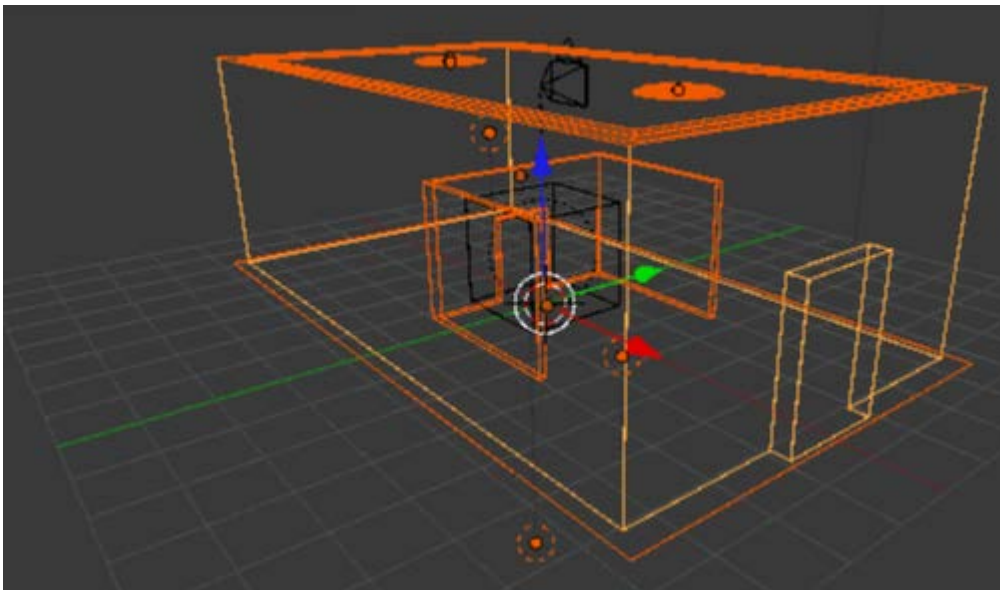
Lo necesitamos para tomarlo como origen en el escalado, pero eso sólo funciona si el plano del suelo de la arquitectura también está con valor **Location Z= 0.000**.

Ahora:

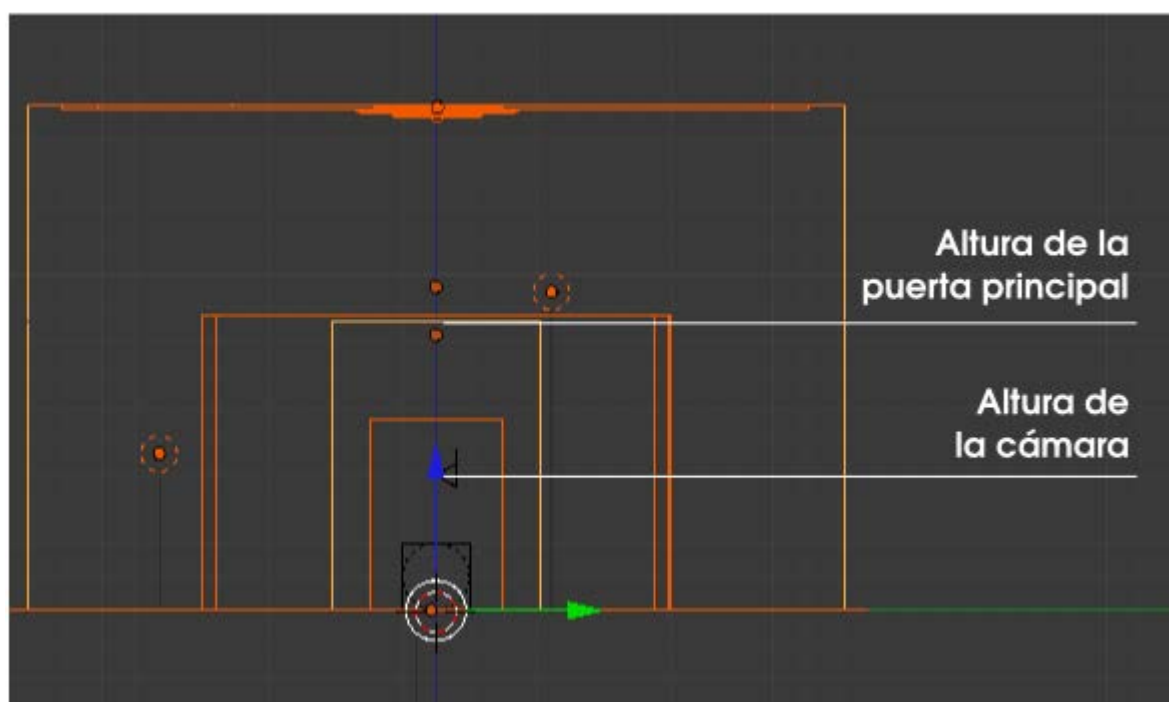
- Cambiamos el **Centro de pivotaje** a **Cursor 3D**.



- Seleccionamos los dos objetos de **CameraFPS** y después usamos **Seleccionar/Inverso**.

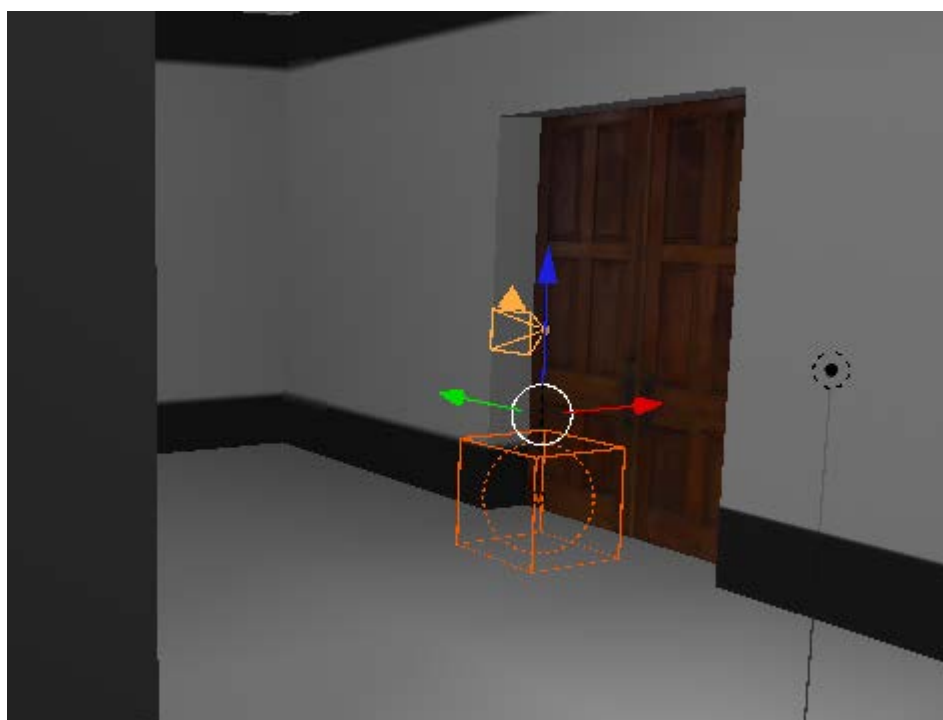


- Nos colocamos en un punto de vista ortográfico ("**NumPad 5**") y en sombreado **Alambre** ("**Z**") para hacer el escalado ("**S**"). La finalidad es controlar que la cámara quede a una altura correcta respecto a las puertas.



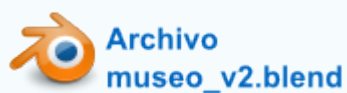
Sólo nos queda:

- Arreglar el tema de la iluminación volviendo a estudiar los valores de **Energía** de las lámparas
- **Girar el artilugio 90° en el eje Z ("RZ90")** y colocarlo de espaldas a la puerta principal.
- No olvidar volver a poner el **Centro de pivotaje** a **Punto medio** para seguir trabajando.



Analiza y estudia el archivo .blend

Usa este .blend para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.



Más mapeados

Si somos sinceros hay que decir que el tema de mapear es casi un oficio en si mismo. En un equipo de trabajo hay auténticos especialistas en sacar el máximo partido a todos los entresijos de esta técnica.

No es la intención de **Blender: 3D en la Educación** plantear técnicas secretas ni nada parecido, pero vamos a ahondar un poco más en el tema de los mapeados, sobre todo cuando están relacionados con la creación de simulaciones.

Proyectar desde la vista

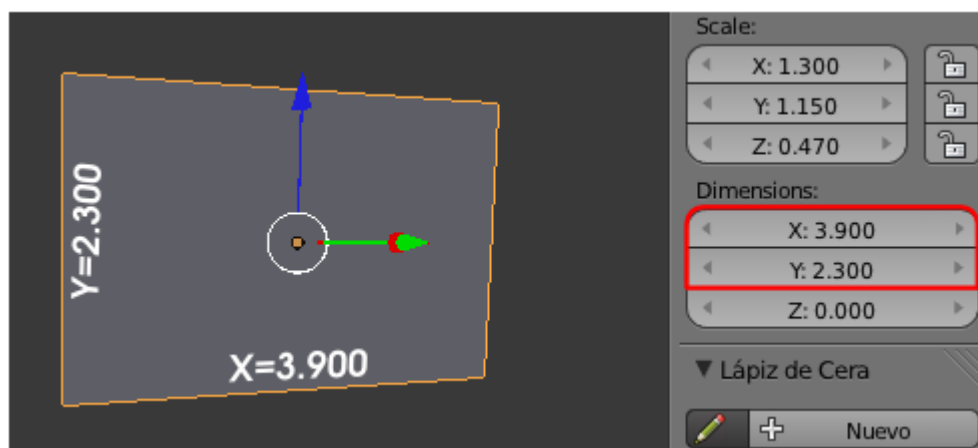
Conocemos la opción **Desenvolver** a la hora de desplegar las caras para un mapeado. Pero esa es sólo una de muchas otras opciones. Una de las más útiles es **Proyectar desde vista**. Si desplegamos de ese modo mientras nos encontramos en un punto de vista frontal, el recurso es tremendamente útil.

A nuestro hipotético museo le vamos a añadir el primer cuadro.

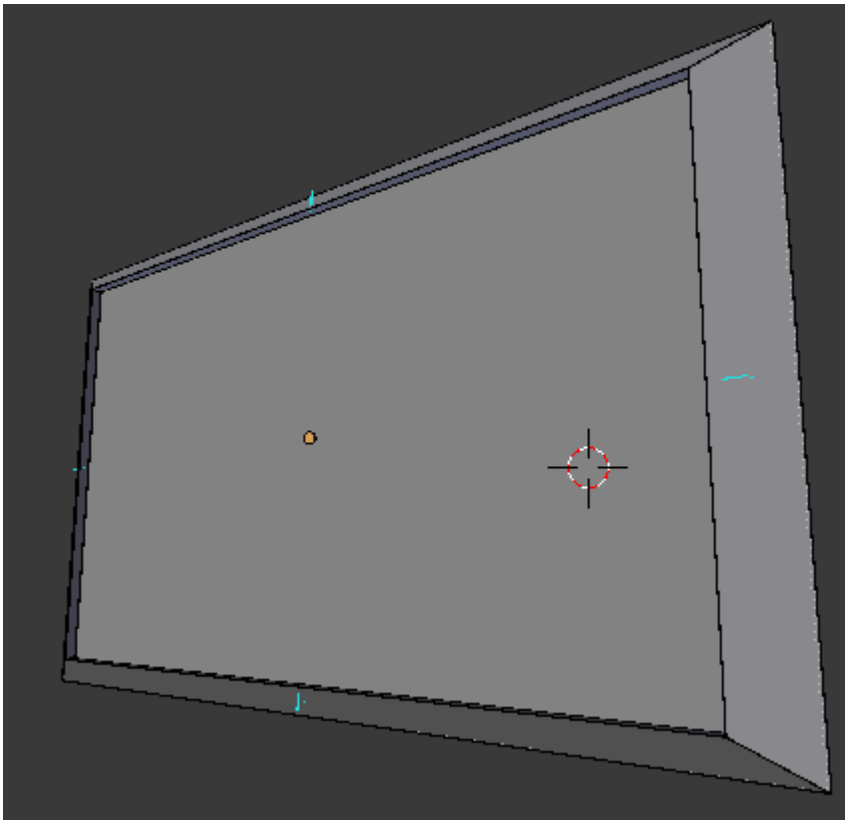


Su modelado **no tiene material** y es verdaderamente sencillo:

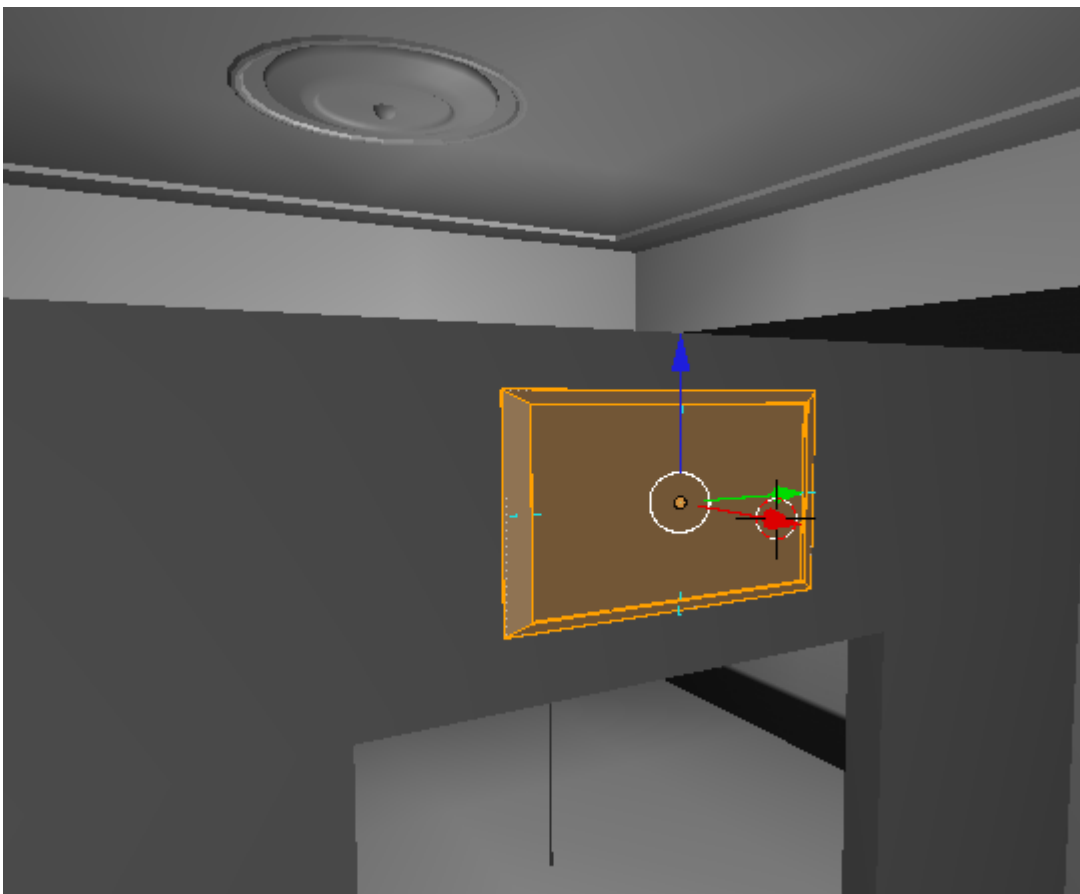
- **Un lienzo** (las medidas del plano conviene que sean proporcionales a 49x83 porque son -en centímetros- las de la obra *Viejos comiendo sopa* de Goya. Nos vale **X=3.900**, **Y=2.300**)...



- Partiendo de la selección de la cara del plano el modelado también tiene **un marco** a base de muy pocos polígonos (*lowpoly*) con algunas **extrusiones** ("E") y **escalados** ("S"). Con desplazamientos adecuados en los vértices esto es lo que tenemos (hemos hecho visibles las **Normales**, una buena costumbre cuando estamos mapeando).



La imagen anterior nos informa de que la primera **extrusión** ("E"), la que sale perpendicular del lienzo ha quedado con la **Normal** invertida. Así que seleccionamos todas las caras ("A") y usamos el menú **Malla/Normales/Recalcular hacia afuera**. Una vez colocado encima de la puerta de la galería interior, es posible que determinemos **escalarlo** ("S") de una manera global para conseguir que la puerta parezca más ancha.



El método para el mapeado en la cara donde plasmar la pintura es ya conocido... Estas son las imágenes que vamos a usar.



Viejas comiendo sopas y Marco dorado



Viejos comiendo sopa //

Autor: Francisco de

Goya // Licencia:

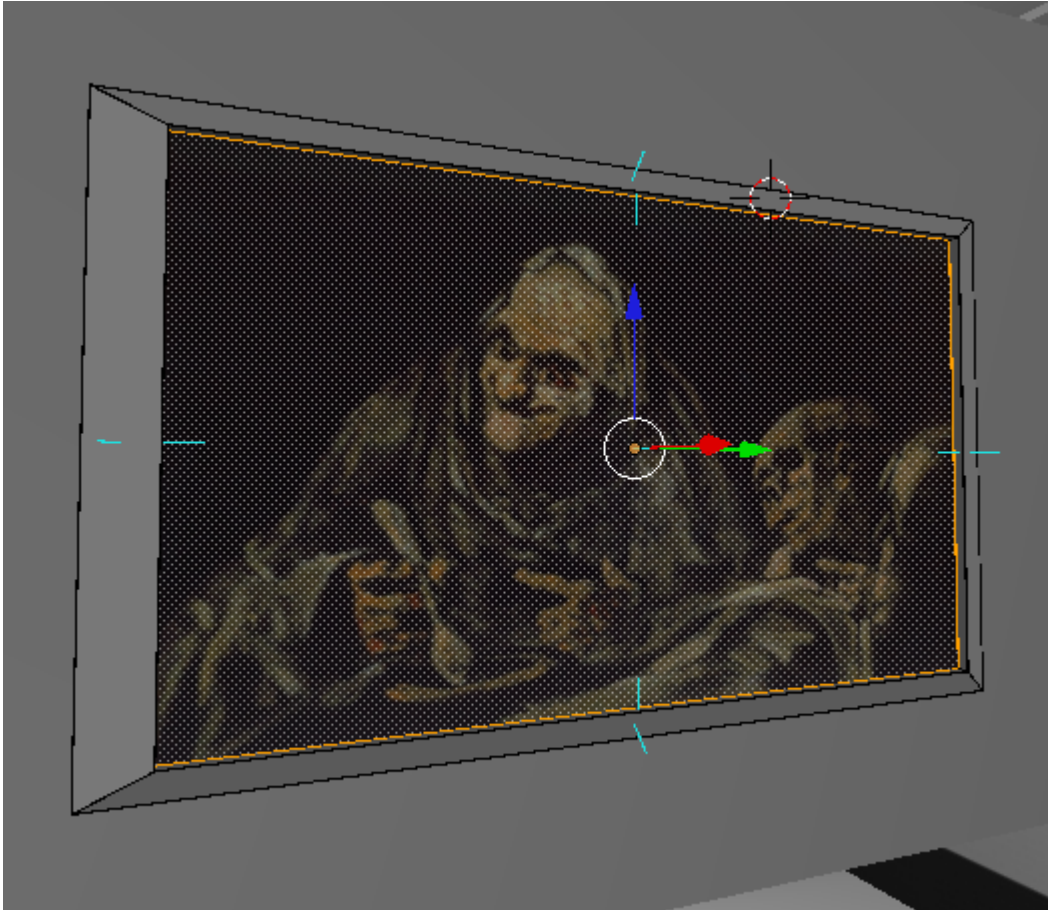
Dominio público

Marco dorado //


Licencia: Dominio

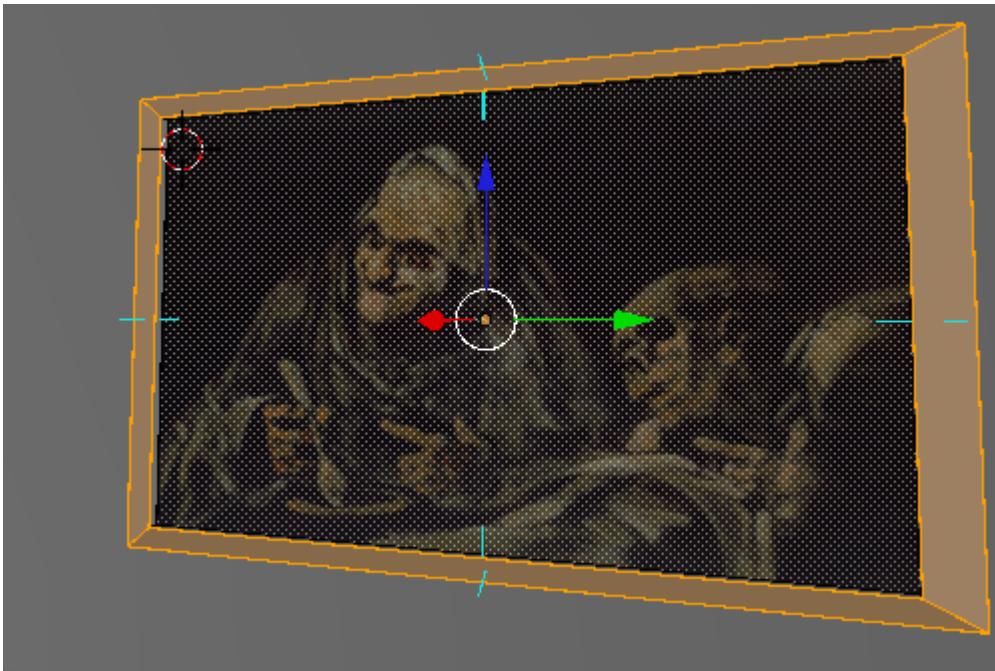
Público

Este es el resultado del mapeado sobre la cara central del cuadro.




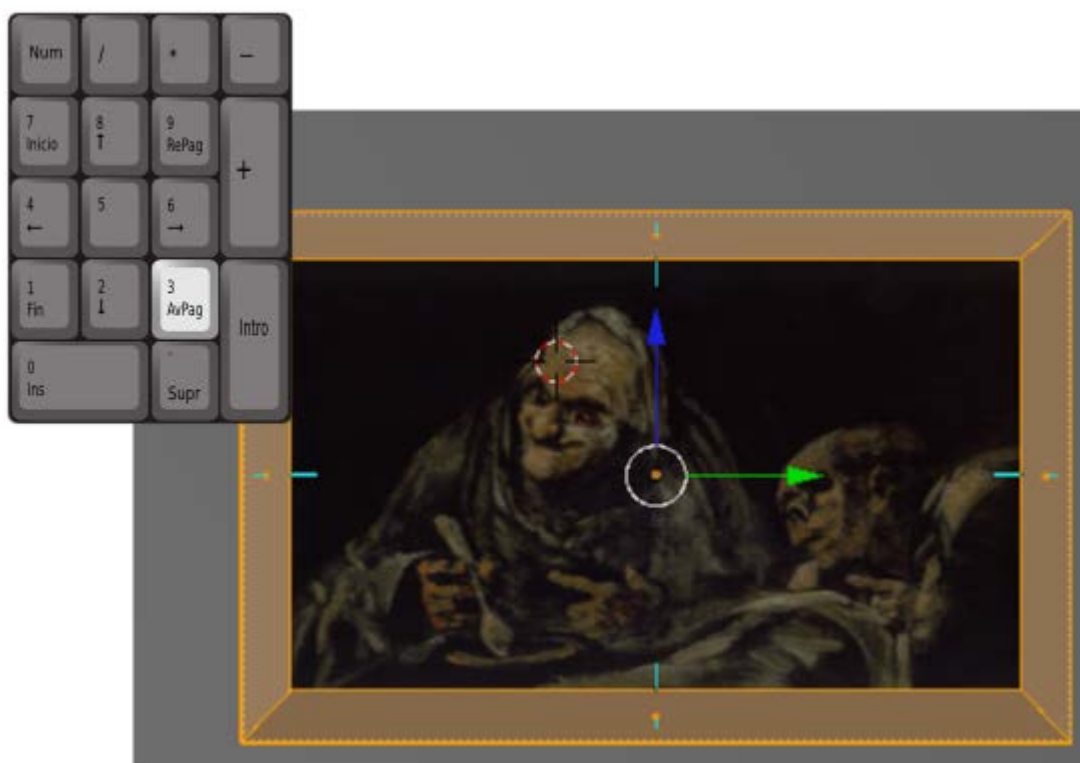
Así que vamos a ver el mapeado correspondiente al marco.

De todos los dos bucles de caras correspondientes al marco seleccionamos  ("Alt_Clic derecho") el más frontal dejando fuera de la selección el que sale perpendicular del plano del lienzo.

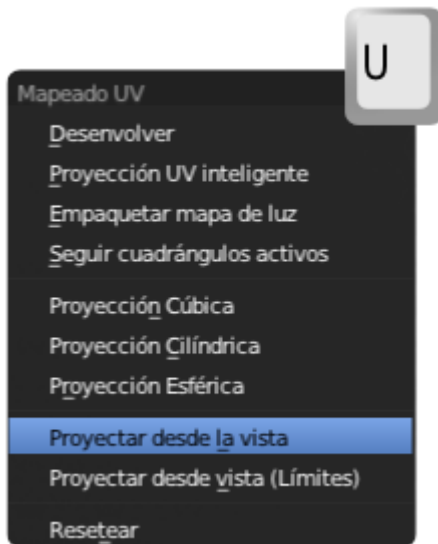



Nos vamos al entorno de trabajo **UV Editing**:

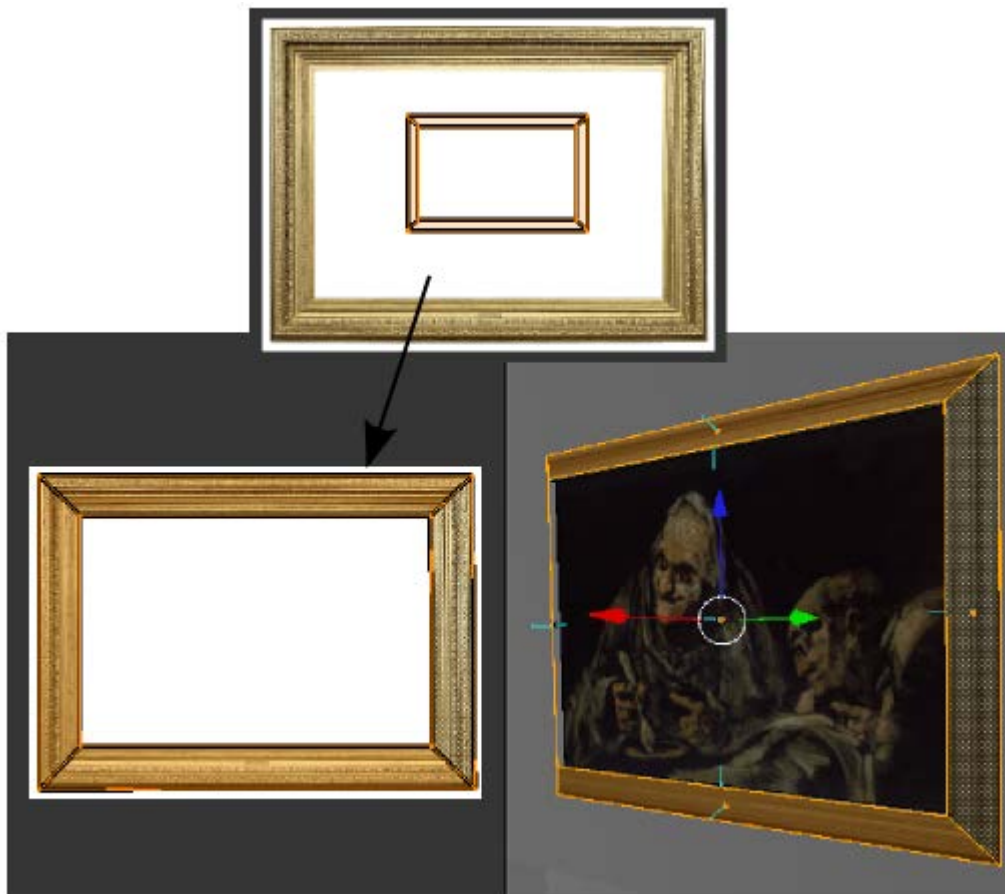
- En el **Editor UV**  cargamos la imagen *marco.png* que hemos descargado a nuestro disco duro.
- En el editor **Vista 3D** nos colocamos en un punto de vista frontal al cuadro (en nuestro caso "**NumPad 3**"). No es necesario que sea ortográfico aunque eso puede ayudar a tener una mejor visión de la técnica que estamos usando.



Es el momento de dar la orden **Malla/Desplegar UVs/Proyectar desde la vista** (o el atajo "**U**") desde el editor **Vista 3D**.



Esto obliga a Blender a proyectar sin deformar, y eso se adapta por completo a nuestras necesidades en el **Editor UV** . Después sólo nos preocupamos de adaptar los vértices para que el mapeado sea correcto.



En el motor de juegos ("P") presenta una sensación muy volumétrica sin apenas polígonos.




Ayuda visual



Vídeo-tutorial Proyectar desde vista



Los materiales

Ahora asignamos **Material**  al objeto pero para que quede realmente bien atenderemos a estos consejos. La malla tiene tres materiales asignados a sus correspondientes caras:



- **Lienzo.** Tiene que tener la opción **Textura en caras** activada o no saldrá en la simulación ("P"). El valor de la **Intensidad de Especular** debe ser muy bajo para que no refleje luz.
- **Marco_con_uv.** Exactamente igual que la anterior aunque aquí la **Intensidad de Especular** puede ser algo más alto al tratarse de un material brillante.
- **Marco_sin_uv.** Es el material asignado al bucle de caras que sale perpendicular al lienzo. Como no tiene mapeado hay que darle un color (**AC9E00** puede ser una buena opción).

De esa manera en la simulación presenta este aspecto.





Analiza y estudia el archivo .blend

Usa este .blend para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.



Archivo
museo_un_cuadro_uv.blend

Alfa por las dos caras

Al asignar un mapeado UV sobre un objeto sin material, Blender se comporta de un modo muy inteligente; por ejemplo, ya hemos visto cómo Blender entiende que si el objeto no tiene material, pero se le ha mapeado una textura, lo lógico es que se vea en el motor de juegos y por tanto le asigna por defecto la opción **Textura en caras**.

Eso mismo ocurre cuando lo que mapeamos es una textura con transparencias. Vamos a hacer una bonita lámpara de araña para nuestro museo virtual.



Archivo
museo_un_cuadro_uv.blend

En su momento, en **Material didáctico: Galileo en Pisa**, vimos una extraordinaria extensión para importar imágenes. Se comentó que esa importación deja el material correctamente tanto para **Blender Estándar** como para **Blender Game**. Lo que aquí vamos a ver es el proceso manual por el cual somos nosotros los que configuramos todo; sólo así comprenderemos qué hace la extensión **Import Images as Planes** y cuáles son las opciones que debemos manipular si deseamos un resultado diferente.

Esta es la imagen que usaremos (es un PNG con fondo transparente).

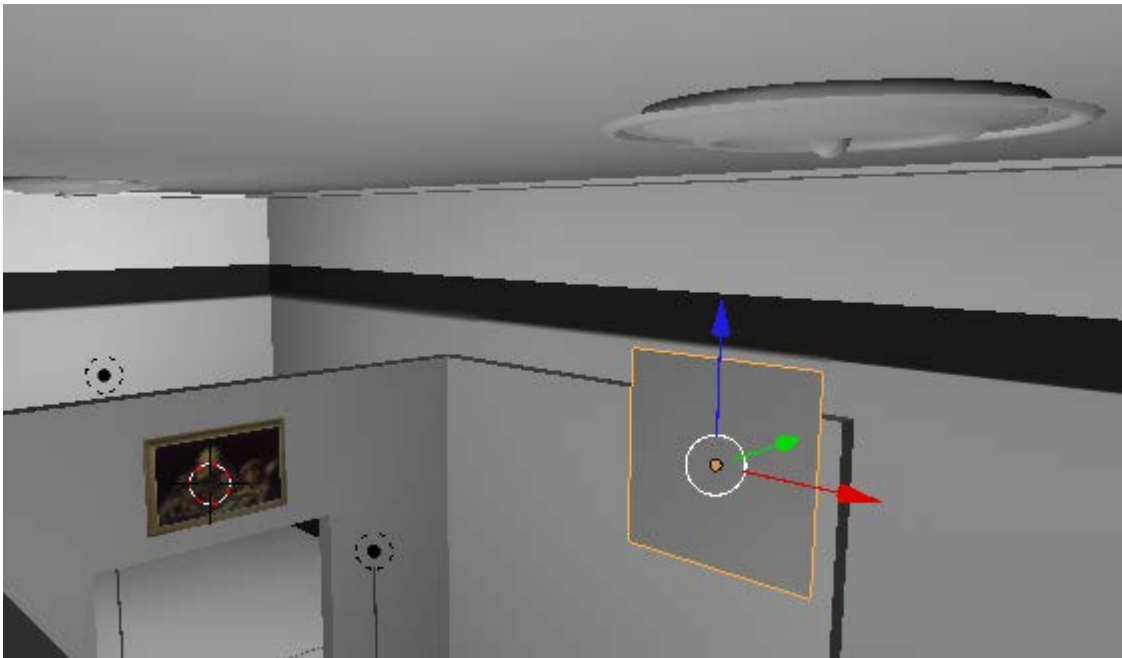


Módulo para lámpara de araña

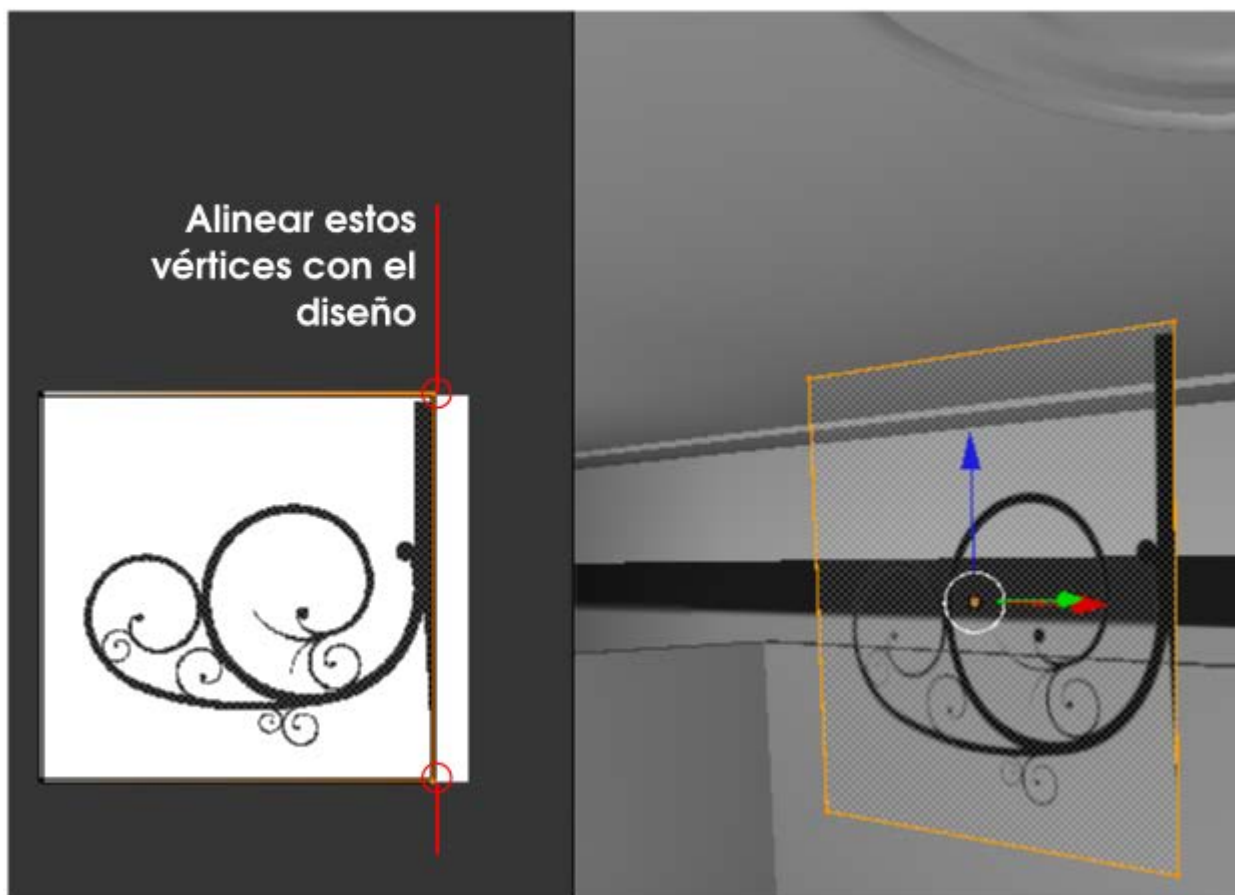


Módulo para lámpara de
araña // Autor: Joaclint //
Licencia: CC-SA-BY
(Creative Commons)

Sacamos un plano a escena y lo **rotamos 90° en X ("RX90")**. Si es necesario lo **escalamos ("S")** para que vaya colocándose cerca de su destino final. Recordamos que **no debe tener material asignado**.



En ese plano, y según el método ya aprendido mapeamos la imagen guardada en nuestro disco duro.



Si activamos la simulación, todo parece perfecto pero no lo es. Estos son los problemas:


- Los contornos no son de calidad. Eso se soluciona fácilmente activando la opción **Premultiplicar** en el cuadro **Propiedades** ("N") del **Editor UV** (no del editor **Vista 3D**).




- Si hacemos órbita para mirar la parte de atrás del plano vemos que la imagen no se mapea por esa cara. Y este es el problema que no tiene solución si el objeto carece de material.



Configurar el material

Comenzamos por darle un **Material**  al plano y configurar a mano. En ese momento todo parece irse al traste... (el mapeado desaparece).

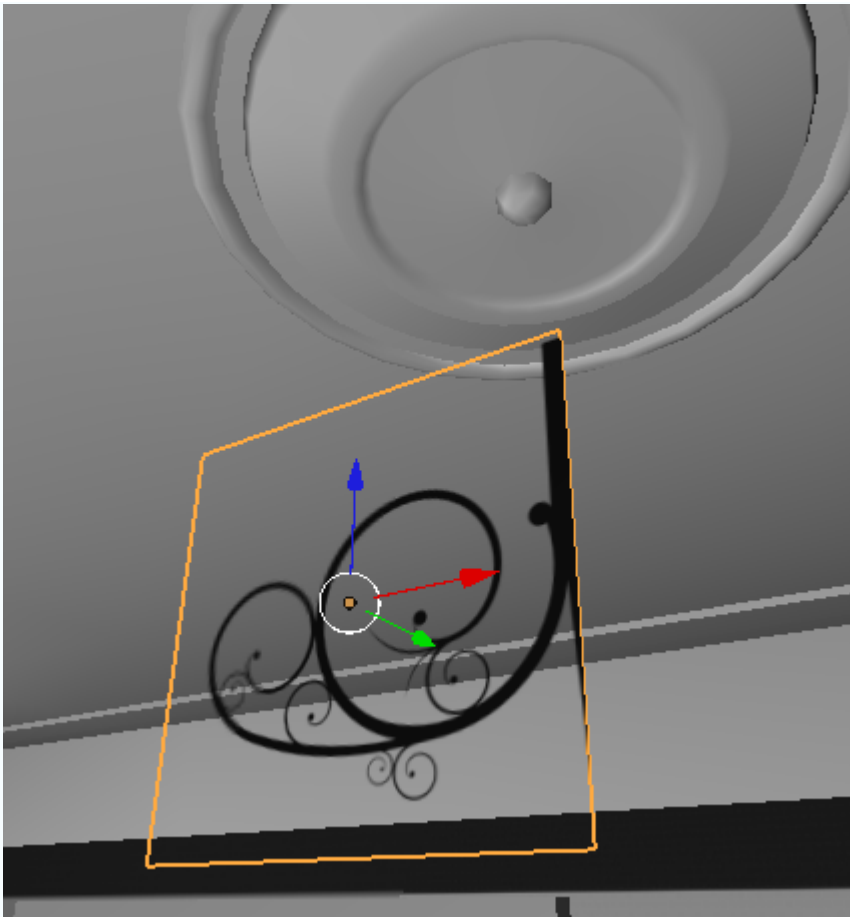
Lo primero, como siempre, es activar **Textura en caras** para asegurarnos que se renderiza en la simulación ("P").

Como estamos en la modalidad de trabajo **Blender Game** en el panel **Material**  aparece una botonera llamada **Opciones de juego**:

- Activamos la transparencia con **Mezcla alfa: Fusionar alfa**.
- Vemos que la opción **Desechar caras traseras** está activada; lo que procede es desactivarla para que se renderice la textura por las dos caras en la simulación ("P").



No estaría mal descender casi a **0.000** la **Intensidad** de **Especular** para que quede casi mate.



Ayuda visual





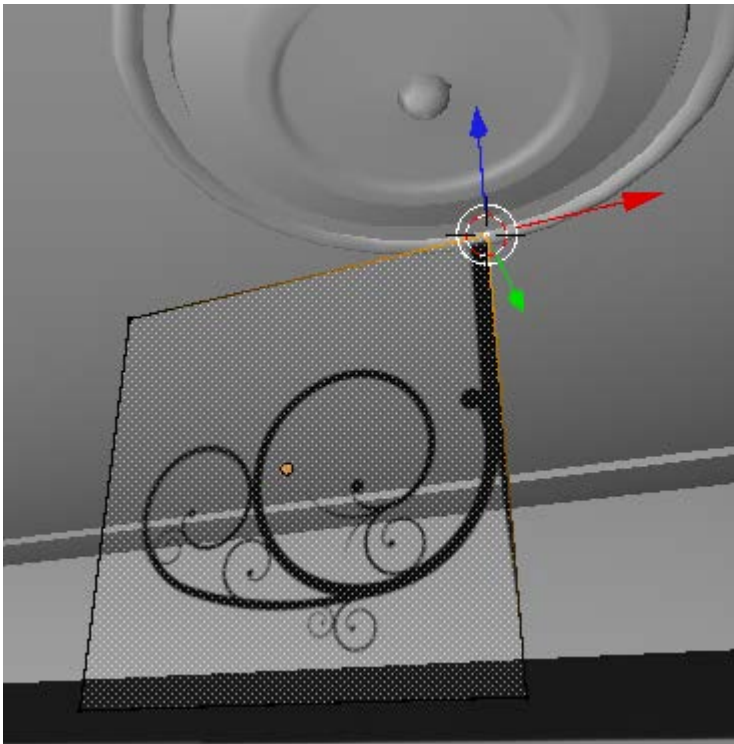
Video-tutorial Alfa por las dos caras





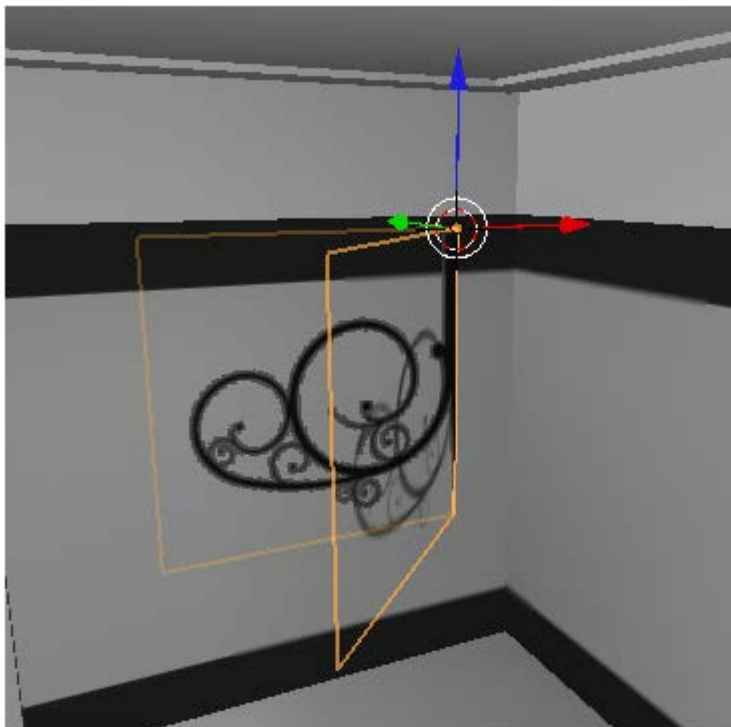
En realidad lo que queda ahora no es un trabajo de mapeado sino de modelado pero damos un breve resumen del método. Queremos que este mapeado se repita ocho veces para conseguir la sensación de lámpara de araña.

Esto es lo que hay que hacer:

- Pasamos a **Modo Edición**  y seleccionamos  uno de los vértices de la parte interior y hacemos **Malla/Adherencia/Cursor a seleccionado**.



- Volvemos a **Modo Objeto**  y usamos **Origen/Origen al Cursor 3D** en el cuadro **Herramientas** ("T"). Esto permite controlar los giros sin necesidad de cambiar de **Centro de pivotaje**.
- Seguimos en **Modo Objeto**  para hacer en una sola edición un **duplicado y rotarlo 45° en Z** ("Shift_D RZ45"). También podemos hacerlo en dos pasos con "Shift_D Intro" y después la **rotación de 45° en Z** ("RZ45").



- Seleccionamos el último objeto (debería haber quedado seleccionado) y vamos repitiendo la misma operación hasta completar los ocho brazos de la araña.



No nos alarmamos al ver que sólo el plano seleccionado muestra la transparencia con calidad, es una cuestión de ahorro de recursos a la que no le daremos importancia porque en la simulación ("P") se ve bien.



Es muy interesante el resultado conseguido con tan solo ocho polígonos.

En principio sólo nos queda añadir un cilindro o un cubo y hacer con él la unión entre la lámpara y el techo.



Analiza y estudia el archivo .blend

Usa este .blend para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.



Archivo
museo_lampara.blend

Sombras

Los objetos en las simulaciones que llevamos hechas carecen de sombras arrojadas. Eso les resta realismo y tridimensionalidad.

Vamos a dedicar este apartado a las distintas opciones para conseguir sombras:

- **Prediseñadas**. Imágenes listas para mapear.
- **Simuladas**. Interesante técnica con la que se puede conseguir, por ejemplo, una falsa oclusión ambiental.
- **Iluminación GLSL**. Es una técnica sofisticada capaz de crear **sombras en tiempo real** durante la simulación. Es la opción adecuada cuando el objeto es móvil y la sombra arrojada no puede ser una imagen fija como en los casos anteriores.

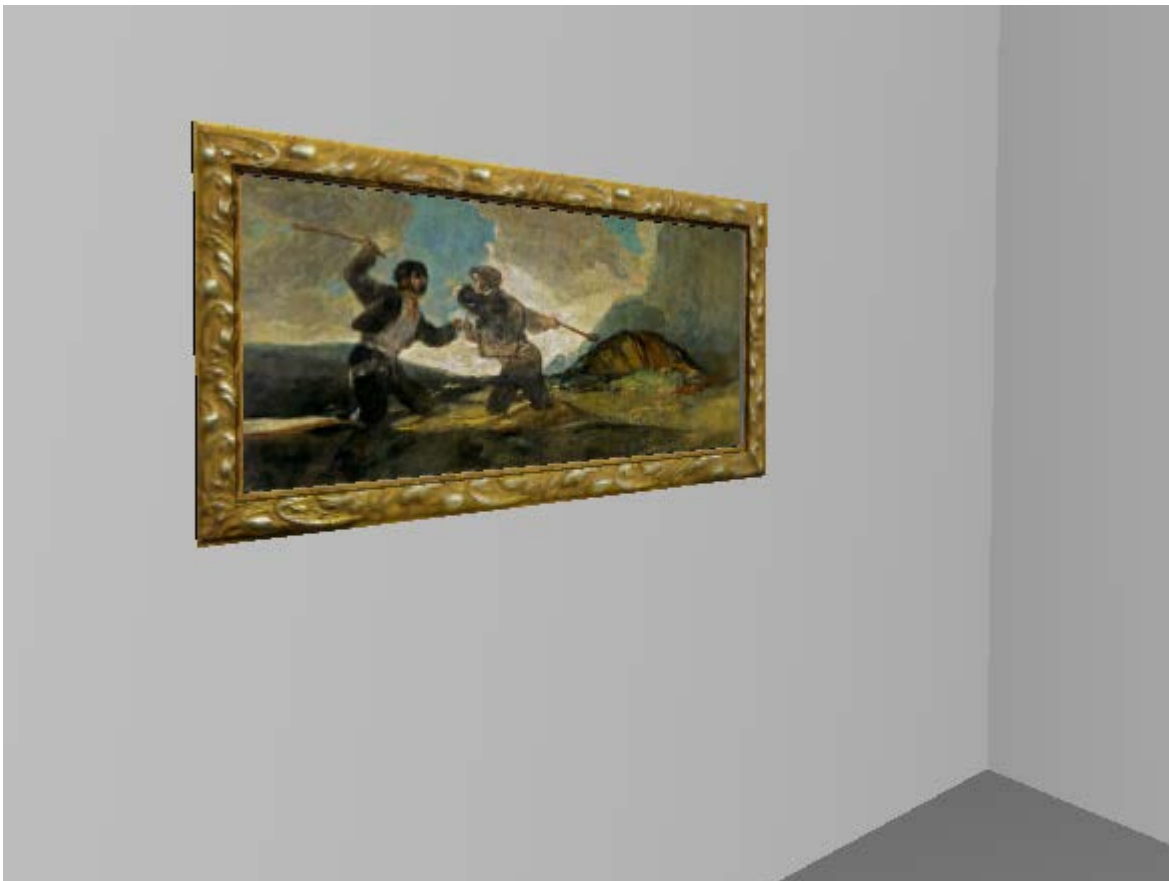
Prediseñadas

Está claro que si se mapea una textura de una pared, es posible que lleve incorporada alguna sombra y prediseñar esa pared con el efecto incluido. Un ejemplo es la siguiente imagen: si mapeamos la textura de arriba, después el cuadro da la sensación de proyectar una sombra.



Este es el método más elemental y se fundamenta en el tipo de mapeado UV que ya sabemos hacer. Tiene el inconveniente de que no es fácil alterar la localización de la sombra aunque con una buena planificación se consiguen bonitos resultados.

Para este tipo de efecto es mucho mejor hacer el mapeado de la sombra en un plano independiente situado entre el cuadro y la pared. Vamos a partir de una escena con una pared y un cuadro que tiene ya mapeada la pintura y el marco.



Añadimos un plano (**Añadir/Malla/Plano**) a la escena. Lo escalamos, rotamos y colocamos tal y como indica la imagen siguiente (lo mejor es usar el modo de sombreado **Alambre** y estar en un punto de vista ortográfico lateral para que el plano se coloque lo más cerca posible de la pared pero sin llegar a tocarla).



Dos asuntos importantes

- No aplicamos material alguno a ese plano. De lo contrario ya sabemos que correrá de nuestra cuenta seleccionar **Mezcla alfa: Ordenar alfa** en **Opciones de juego**, además de activar **Textura en caras** en la botonera **Opciones**.
- Comprobamos que la **Normal** del plano apunte hacia el lado correcto para el mapeado. Activamos la visibilidad de las **Normales** en el cuadro **Propiedades ("N")** en la botonera **Visualización de malla** mientras estamos en **Modo Edición**



En ese plano vamos a mapear una imagen con la sombra. Su principal característica es que se trata de un PNG con transparencia pensado precisamente para que en los lugares adecuados deje ver la pared de atrás.

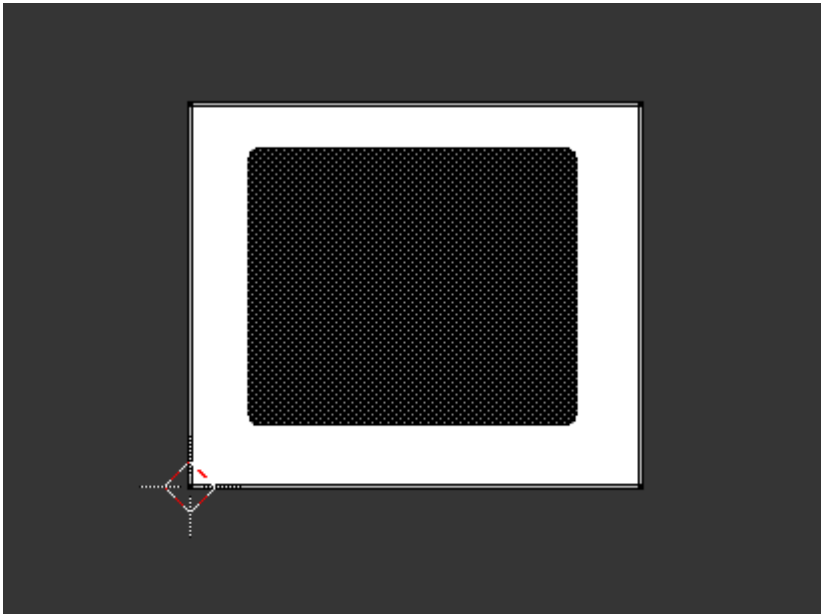


Sombra con transparencia

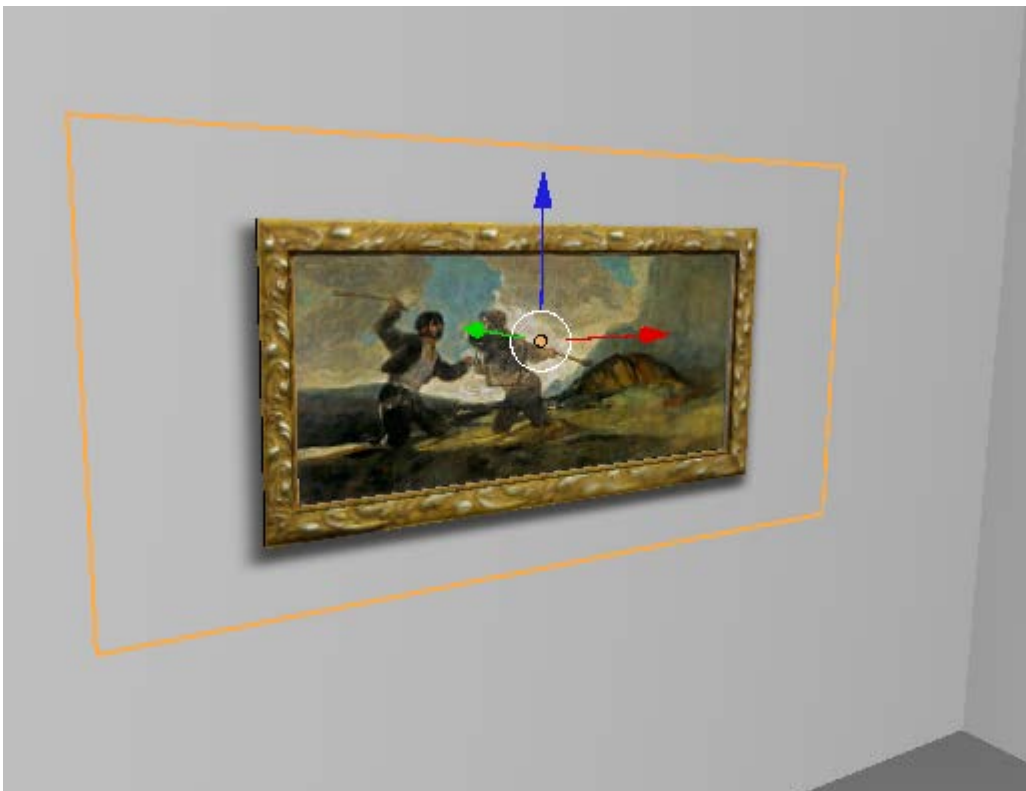


Hacemos el mapeado según lo aprendido hasta ahora. Mostramos un esquema del proceso:

- Pasamos al entorno de trabajo **UV Editing**.
- En el **Editor UV** cargamos la imagen de la sombra que hemos guardado en nuestro disco duro. El contorno difuminado de la sombra no es interpretado de ese modo por el **Editor UV**, pero eso no es un problema para trabajar.

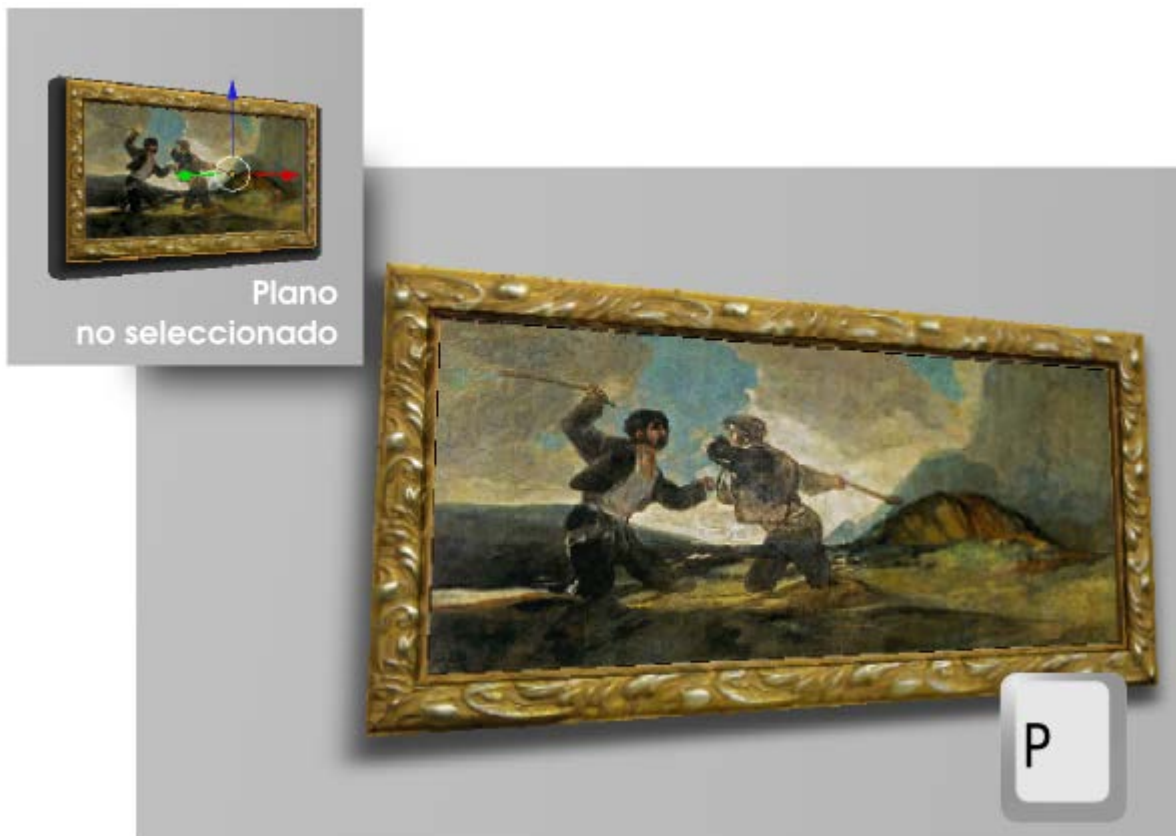


- En el editor **Vista 3D**, en **Modo Edición**, seleccionamos la cara del plano y hacemos **Malla/Desplegar UV/Desenvolver**.
- Ajustamos, aprovechando el sombreado **Textura**, para adaptar la sombra. Estos ajustes los hacemos tanto editando el mapeado UV como escalando el propio plano.



La ventaja de este método es que ahora es posible seleccionar el cuadro con la sombra y desplazarlos sin preocuparnos de nada más.

Cuando no tengamos seleccionado el plano para editarlo, la sombra se mostrará sin grados de transparencia, pero eso no significa nada porque al ejecutar la simulación ("P") no hay ningún problema.



Analiza y estudia el archivo .blend

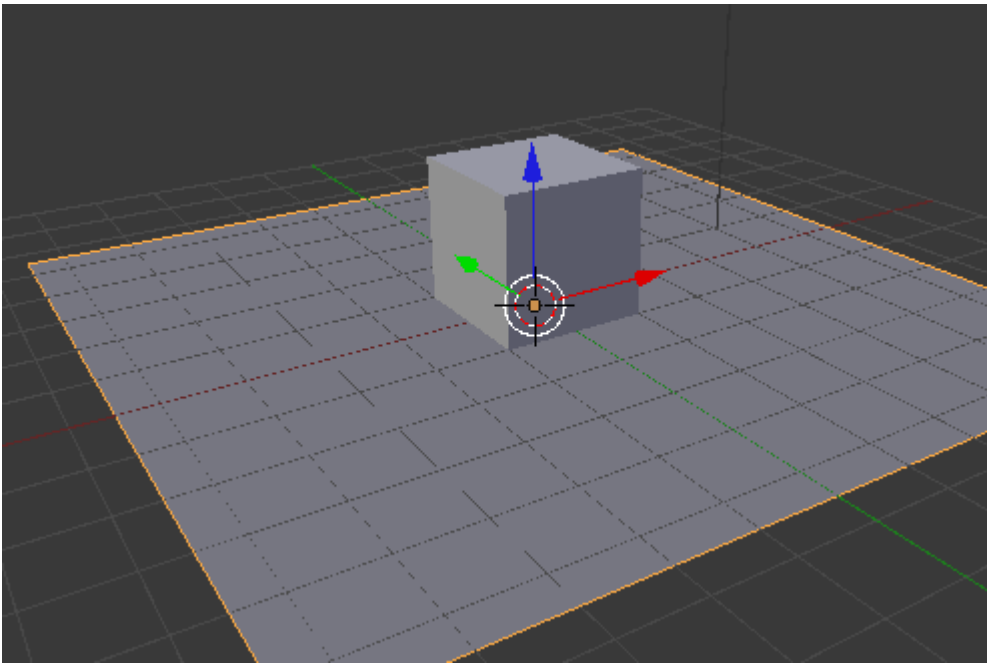
Usa este .blend para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.



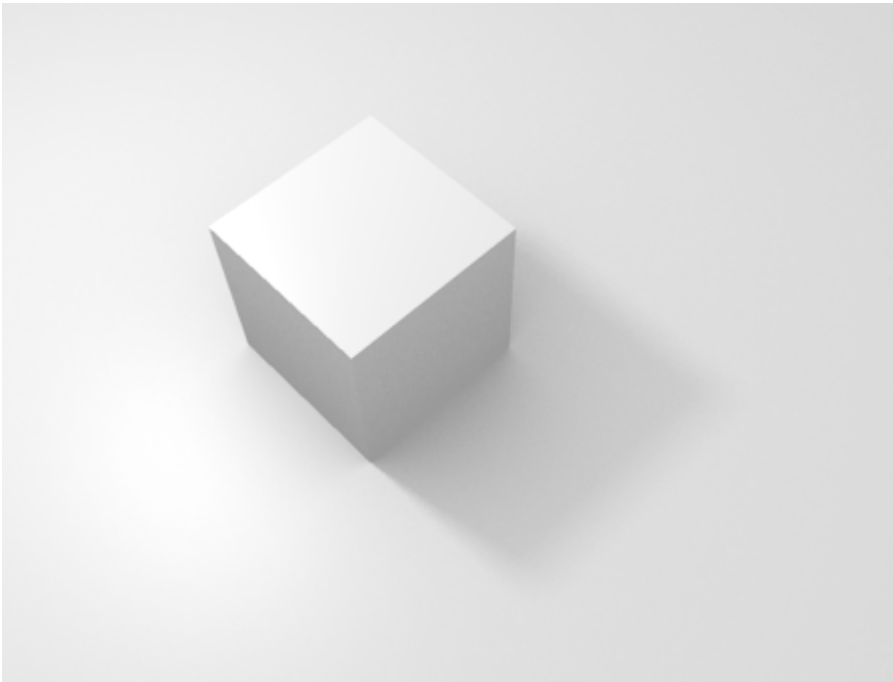
Simuladas

Mucho más interesante que hacer nosotros los gráficos para las sombras es aprovecharse de los recursos que incorpora Blender para crear texturas que se mapean automáticamente tras un renderizado.


Lo vemos en un ejemplo sencillo en el que partimos de una escena de un cubo sobre un plano.

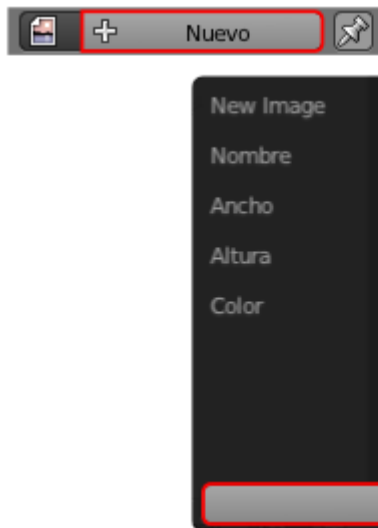


Aquí no es tan importante si el plano tiene o no asignado un material. En el caso de tenerlo, lo único de lo que nos preocupamos ahora es de que **no** tenga activada **Textura en caras** en la botonera **Opciones**.

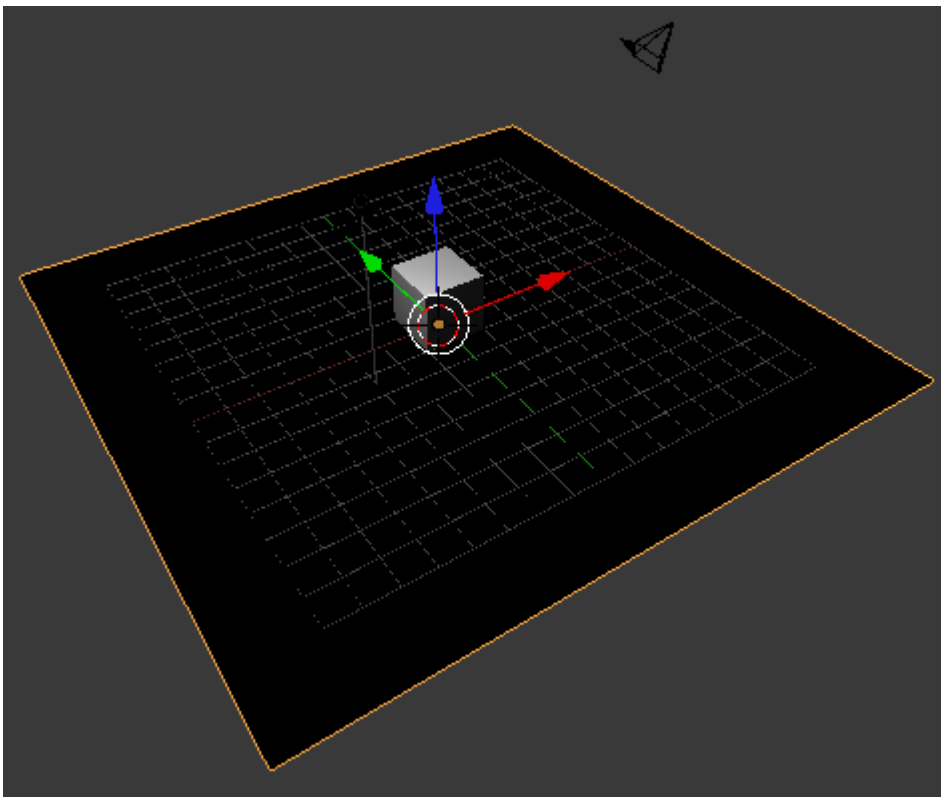



Sería fantástico conseguir un efecto de sombra arrojada como esa, e incluso que se conservara ese tono de oclusión ambiental. ¿Se puede?. Sí, y además con muy poco trabajo. Vamos por orden:

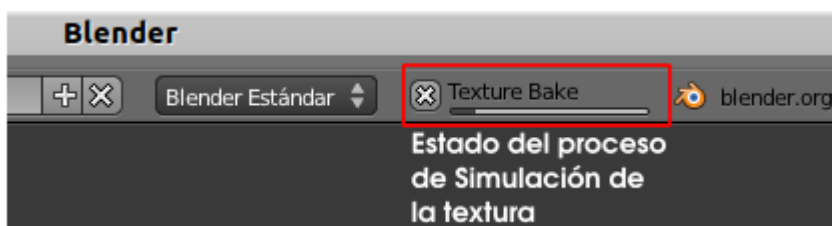
- Nos vamos al entorno **UV Editing**.
- En el **Editor UV**  creamos una imagen (no la cargamos tal y como estamos acostumbrados) pulsando el botón **Nuevo**. En el cuadro que se nos muestra le damos un **Nombre** (*suelo* es una buena opción) pulsamos **Ok**. Con esto se crea una imagen de 1024x1024 completamente negra (si nos fijamos ese es el color por defecto).



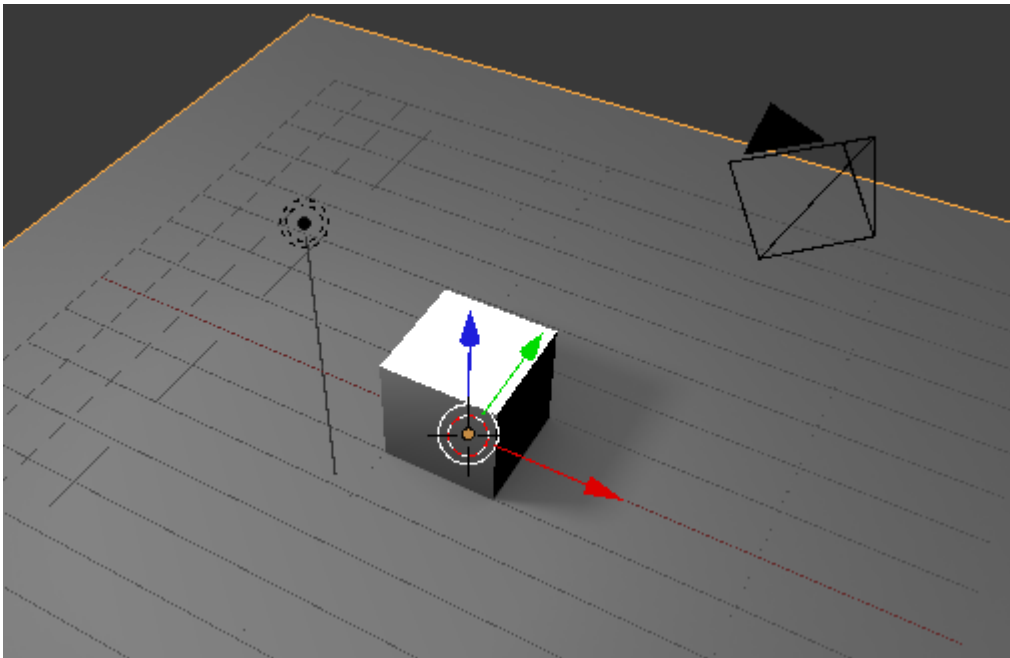
- Como es habitual, le asignamos la imagen desde el editor **Vista 3D** con **Malla/Desplegar UVs/Desenvolver**. El resultado con sombreado **Textura** debe ser algo así.



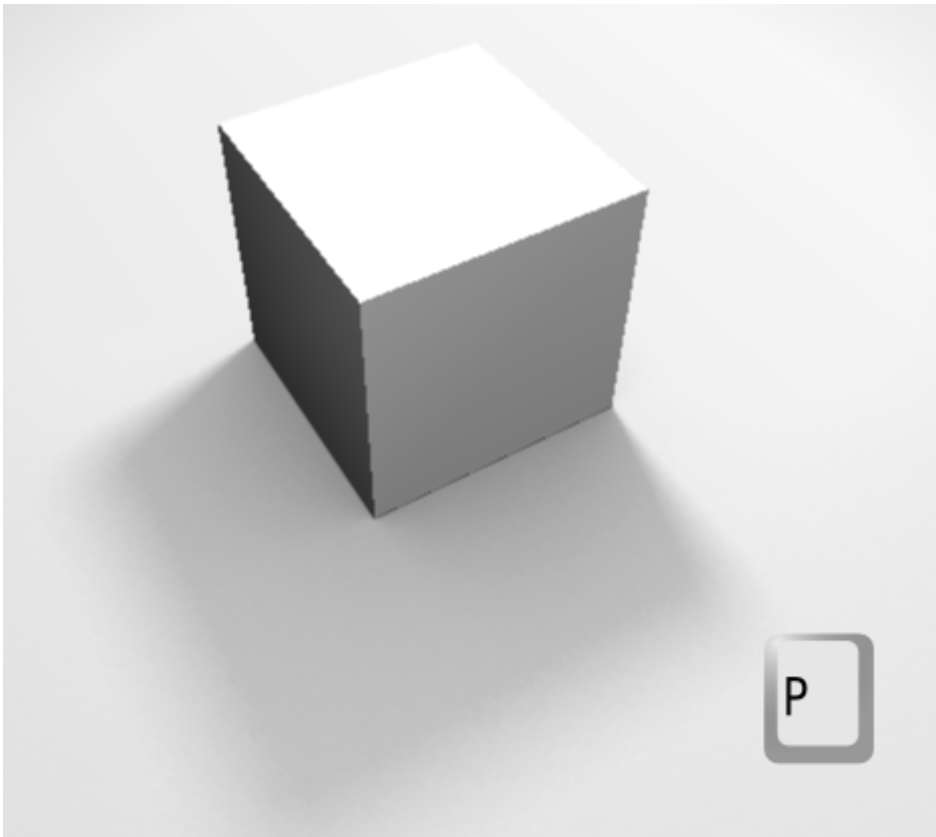
- Regresamos al entorno de trabajo **Default** y nos vamos directos al panel **Render** , buscamos la botonera **Simular** y pulsamos el botón del mismo nombre. El resultado no es instantáneo porque Blender necesita un tiempo para hacer el *render* y asignárselo al plano. Este proceso se muestra en la parte alta de la interfaz



Al final del proceso la textura se asigna al plano de modo automático (no olvidemos estar en sombreado **Textura** para verlo en tiempo real).



Si cambiamos al entorno **UV Editing** confirmamos que la imagen negra se ha sustituido por la textura simulada. Y ese aspecto, a la hora del poner en marcha el motor de juegos, es muy volumétrico (para esta imagen se han añadido más lámparas para evitar caras completamente negras)



Ayuda visual



Vídeo-tutorial Texturas simuladas



Asignar material al plano

Si el plano no tiene asignado un material Blender se ocupa de todo para el texturizado, pero si le asignamos un material para editar colores, especularidad... debemos recordar:

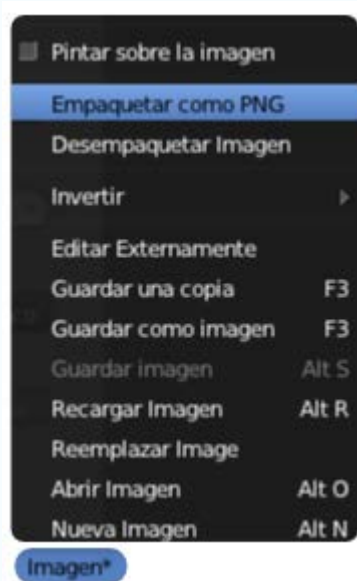
- Activar la opción **Textura en caras** en la botonera **Opciones** del material o a textura no saldrá en la simulación ("P").
- Si queremos repetir la simulación de la textura, la opción **Textura en caras** tiene que estar desactivada.



Empaquetar textura simulada

Como la textura no ha sido cargada desde el exterior sino que se ha creado desde dentro de Blender, la conocida opción **Archivo/Dato externo/Empaquetar en fichero .blend** no funciona y Blender nos envía un mensaje avisando.

La opción correcta es crear el PNG desde el **Editor UV** con **Imagen*/Empaquetar como PNG**. Tras aceptar el mensaje de confirmación el PNG se crea y queda integrado dentro del *.blend*. Blender nos está avisando, mediante un asterisco en la palabra **Imagen** del menú, de que hay imágenes sin crear.





Sombreado GLSL

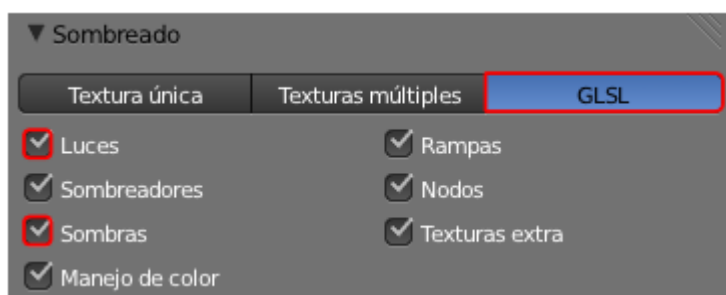
Pero ¿y si los objetos se mueven?. Entonces las técnicas de texturas prediseñadas y simuladas no nos sirven. En esa ocasión hay que echar mano de la técnica más avanzada que se encuentra en Blender para proyectar sombras en tiempo real; es el llamado **sombreado GLSL** (*OpenGL Shading Language*).



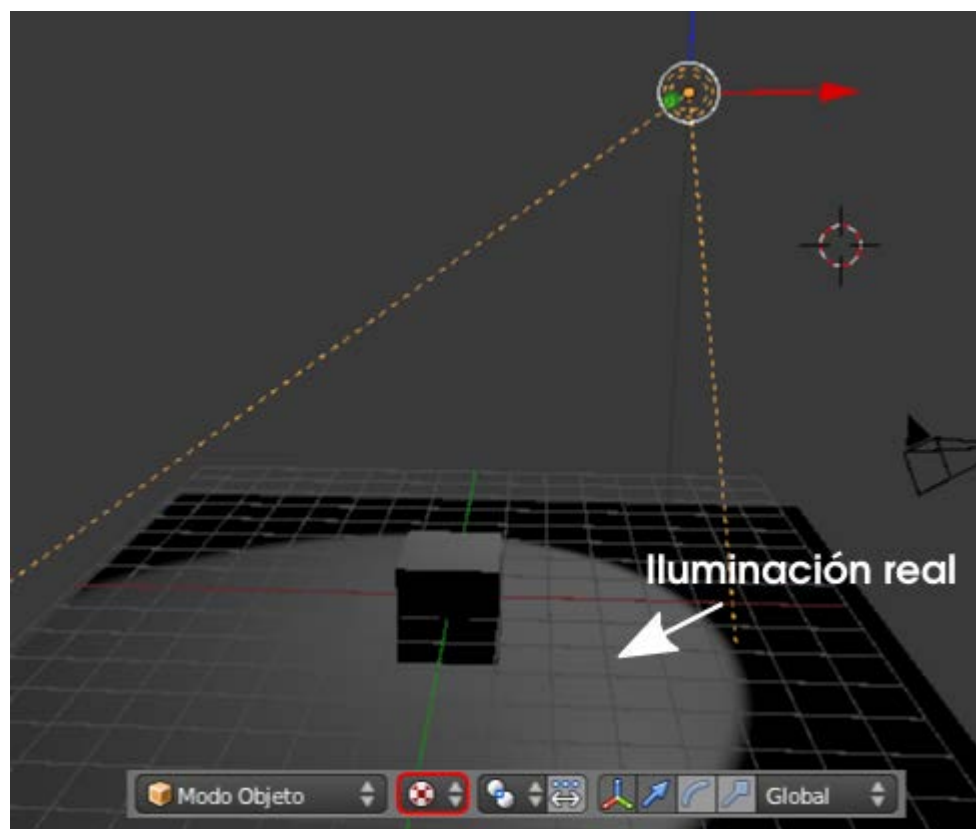
Requisitos para el sombreado GLSL

- Como es lógico debemos estar en la modalidad **Blender Game**.
- El único tipo de lámpara que consigue sombras arrojadas es **Foco**  y es en su panel donde hacemos todos los ajustes.
- En el editor **Vista 3D** sólo se ve en sombreado **Textura**.

La opción se encuentra en el panel **Render** , en la botonera **Sombreado**. Allí, además de activar el sombreado **GLSL**, atenderemos a que estén activadas las opciones **Luces** y **Sombras**.




En una escena sencilla de un cubo sobre un plano, y si hemos cumplido con todos los requisitos descritos, esto es lo que tenemos en el editor **Vista 3D** en sombreado **Textura**, donde se distingue la zona que ilumina el cono de luz sobre el plano.





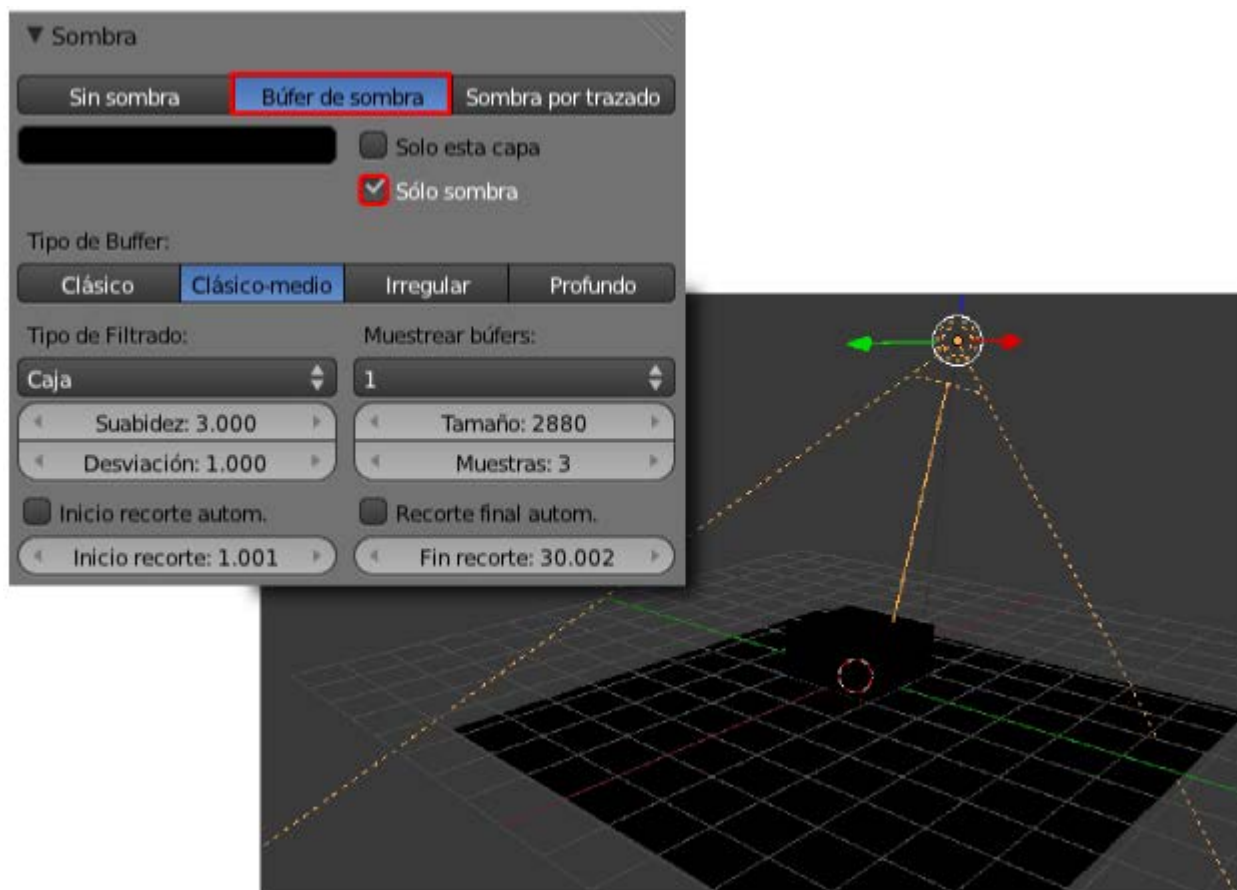
Advertencia

Si hemos cumplido con todos los requisitos y el sombreado GLSL no se nos muestra en el editor **Vista 3D** es que la tarjeta gráfica del ordenador no está capacitada para renderizados 3D de este tipo en tiempo real; nada podremos hacer para solucionarlo.

Para originar la sombra arrojada del cubo sobre el plano activamos en el panel del **Foco**  la modalidad **Búfer de sombra** en la botonera **Sombra**.

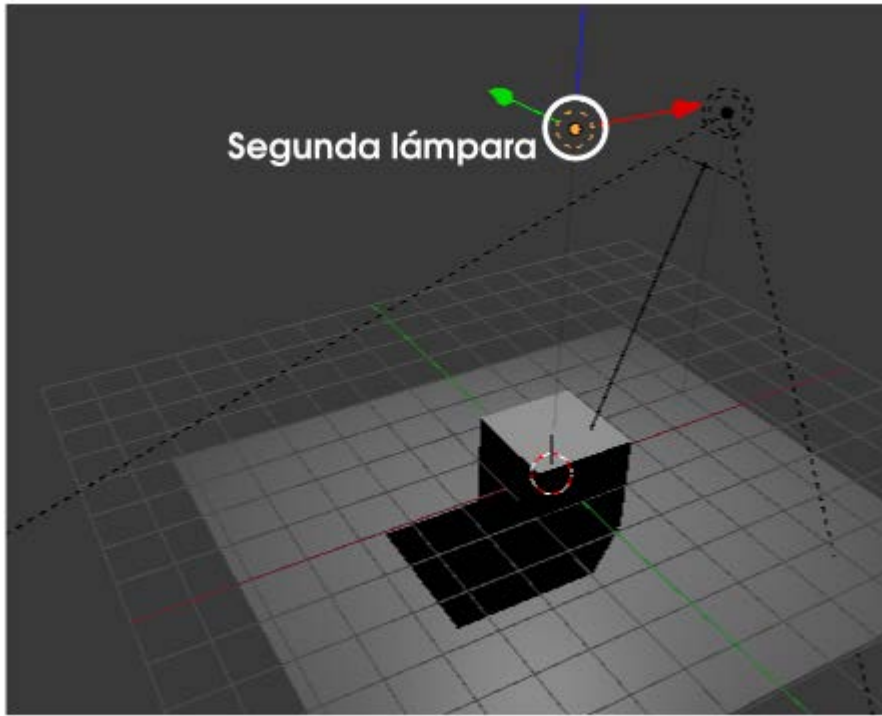




De lo primero que nos deshacemos es del contorno circular luminoso que hace evidente que la lámpara que estamos usando es **Foco** . Activamos la opción **Sólo sombra** en la botonera **Sombra** del panel de la lámpara **Foco** .



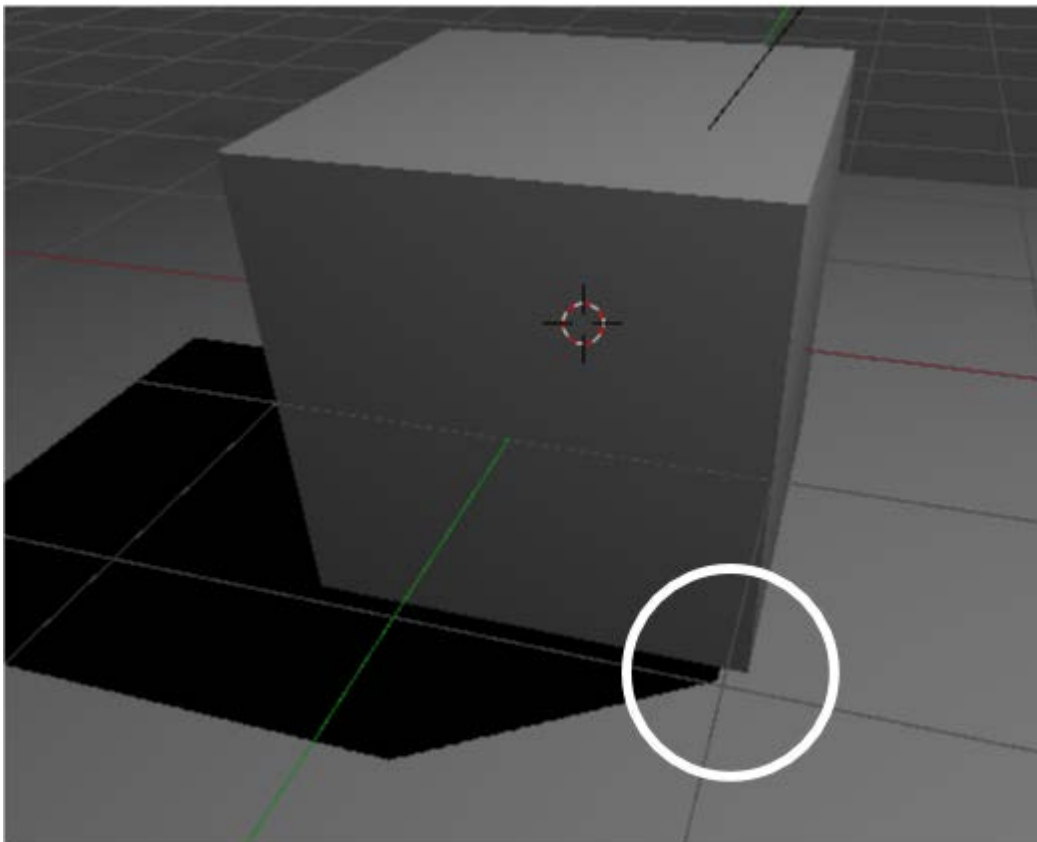
Aparentemente todo se ha ido al traste pero lo que ha pasado es completamente lógico: le hemos dicho a Blender que sólo valore las sombras de esa lámpara por lo que la escena se ha quedado sin iluminación. Nada más que añadimos una lámpara


(**Añadir/Lámpara/Puntual**) en la zona alta el problema se soluciona.

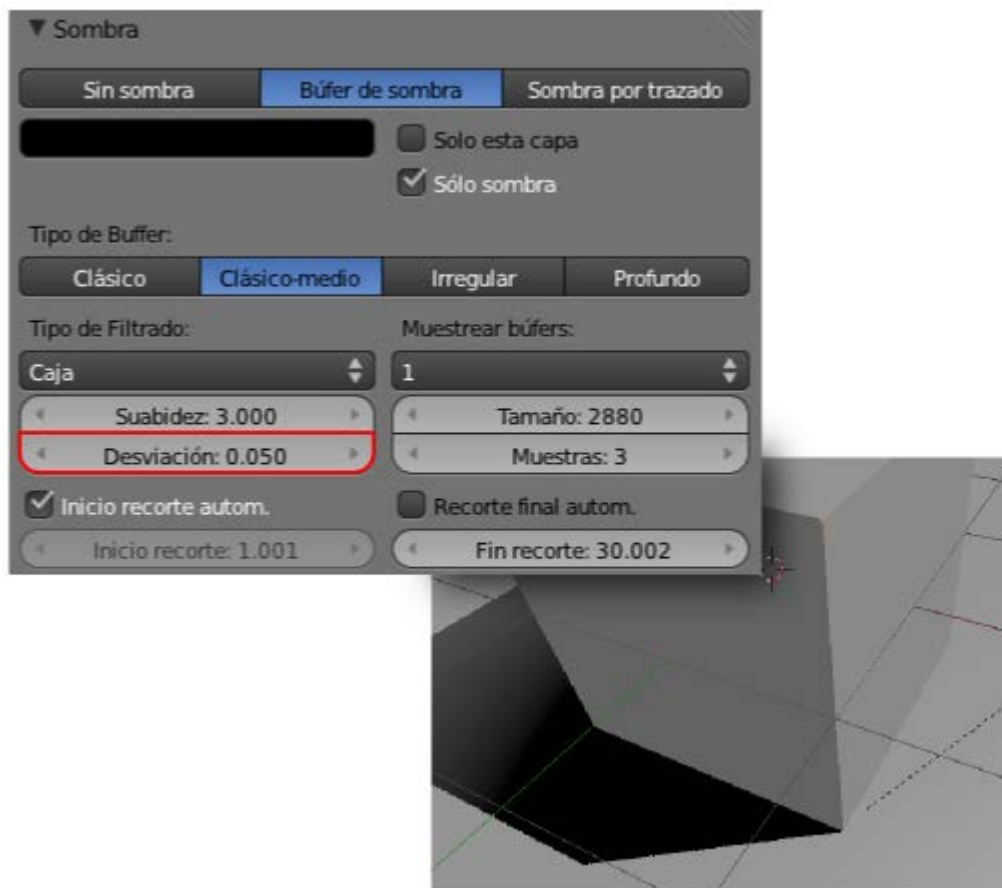


Una iluminación que suele funcionar muy bien es colocar una lámpara **Semiesférica**  cerca del **Foco**  y con la dirección también similar. Esto hace que se produzca una sombra con algo de pérdida de intensidad según se aleja del objeto.

Si nos acercamos a la zona de contacto entre el cubo y la sombra detectaremos cierta imprecisión a pesar de que el cubo y el plano sí se tocan.



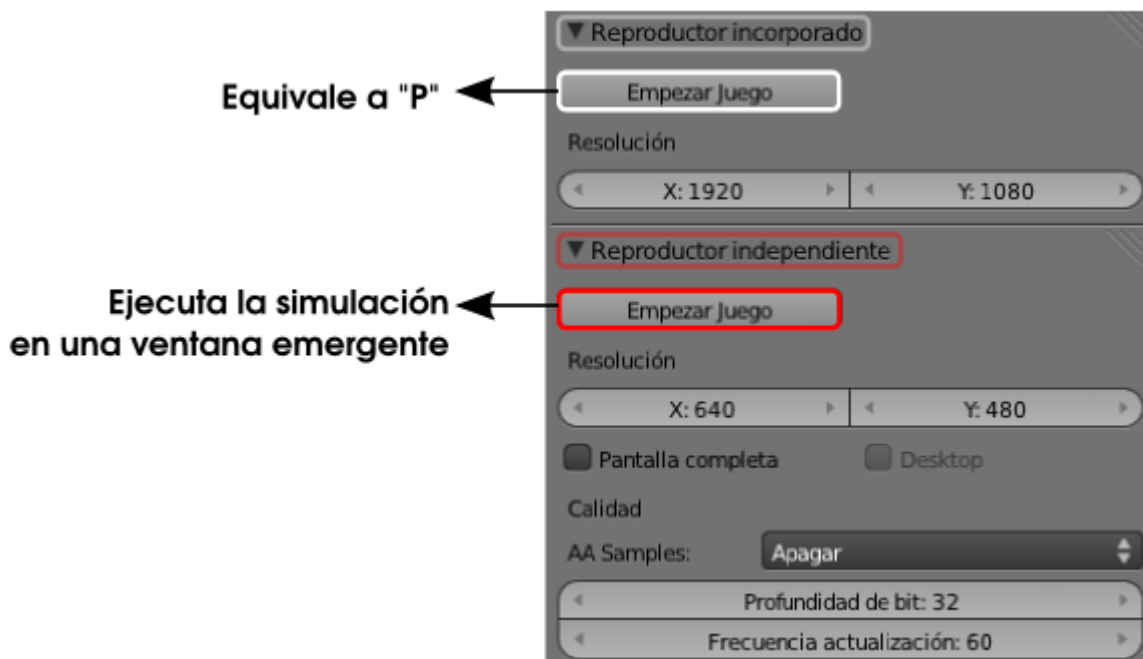
La solución: descender el valor de **Desviación** en la botonera **Sombra** (recordemos que estamos editando la lámpara **Foco** , que es la que origina la sombra)



En ese mismo panel hay un parámetro llamado **Tamaño** que determina la calidad del contorno de la sombra. A mayores valores mejor es el resultado, pero el consumo de memoria RAM se disparará. **3.500** es un valor estándar.

Autoejecutable

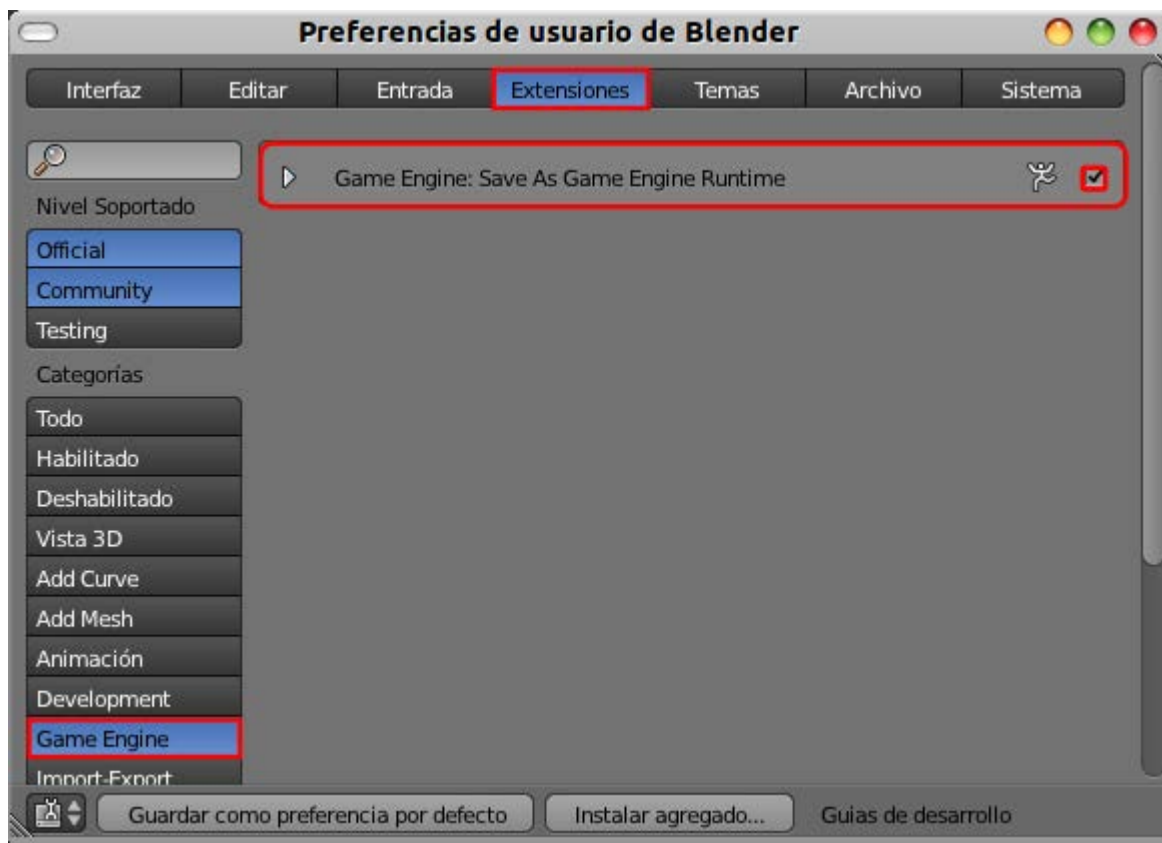
Cuando reproducimos la simulación con el reproductor integrado de Blender se abre una ventana emergente y se ejecuta el programa.



Las opciones de **Resolución** no requieren ninguna explicación siendo realmente interesante **Pantalla completa** y acto seguido activar **Desktop** para que el autoejecutable se adapte a la resolución que tenga el usuario.

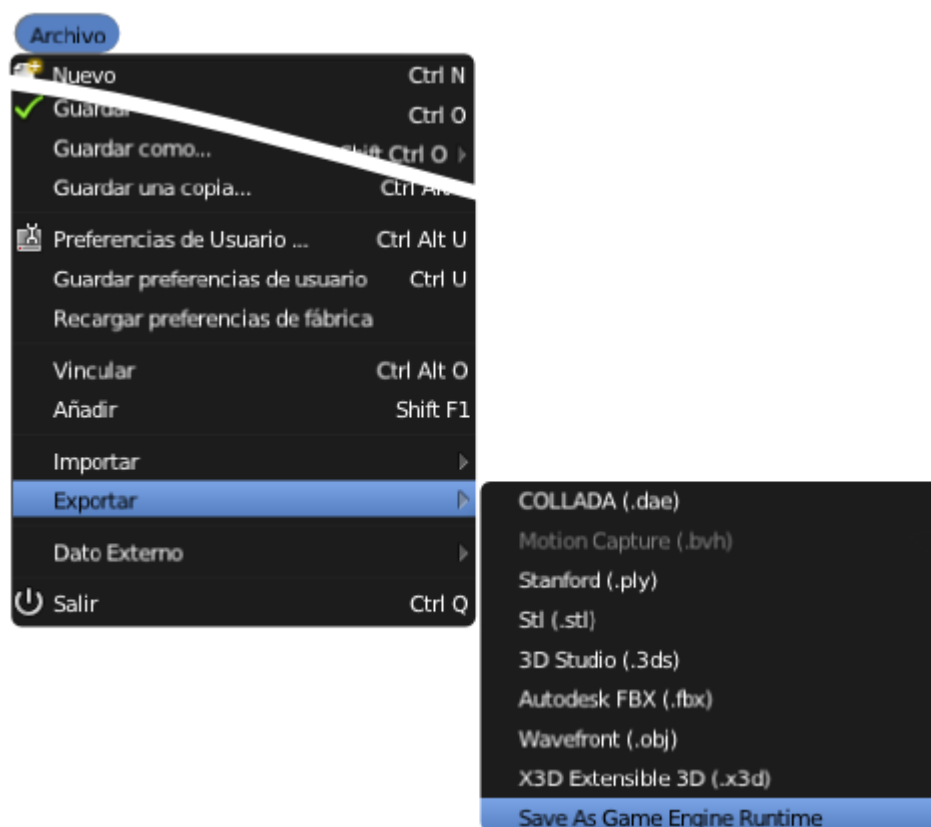
La idea de crear un autoejecutable es conseguir ese mismo efecto pero sin necesidad de lanzarlo desde dentro de Blender; incluso sin la necesidad de tener el propio Blender instalado.

Debemos activar la extensión (**Archivo/Preferencias de usuario**) denominada **Save As Game Engine Runtime**.

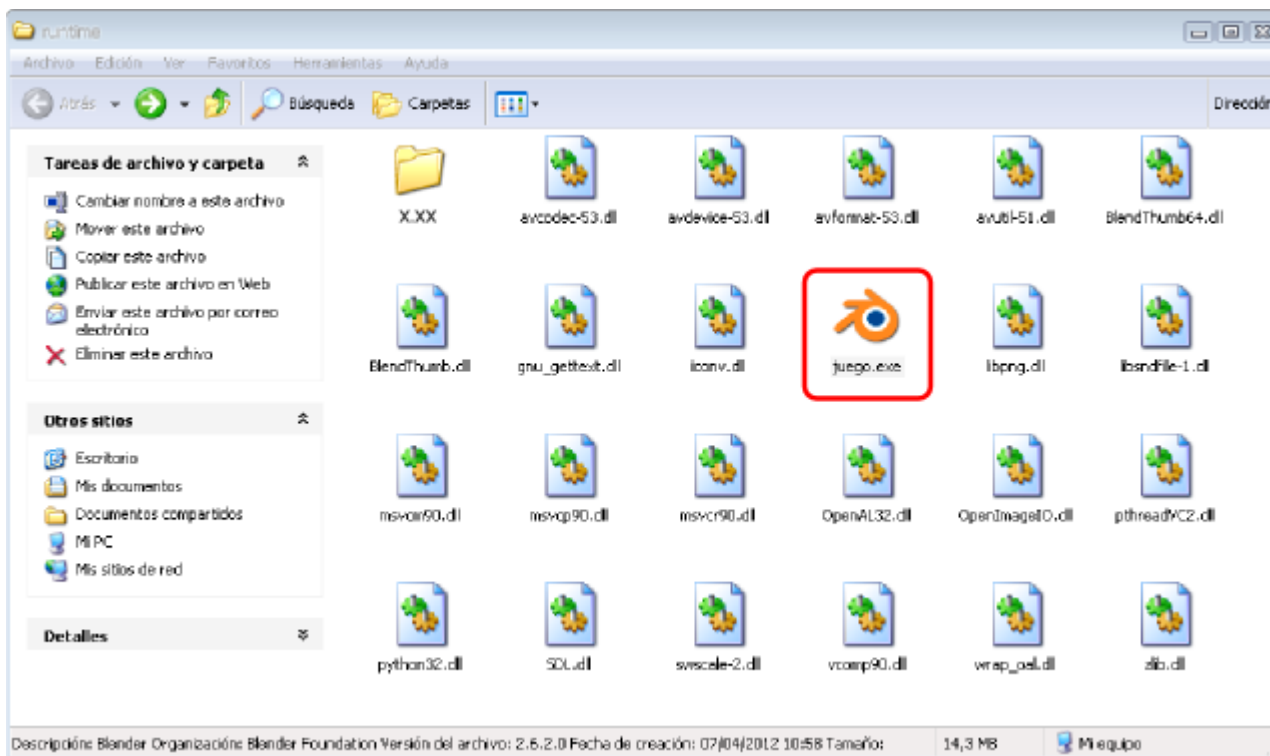


Antes de exportar lo mejor es **crear en algún lugar una carpeta** para este fin, por ejemplo en el *Escritorio*.

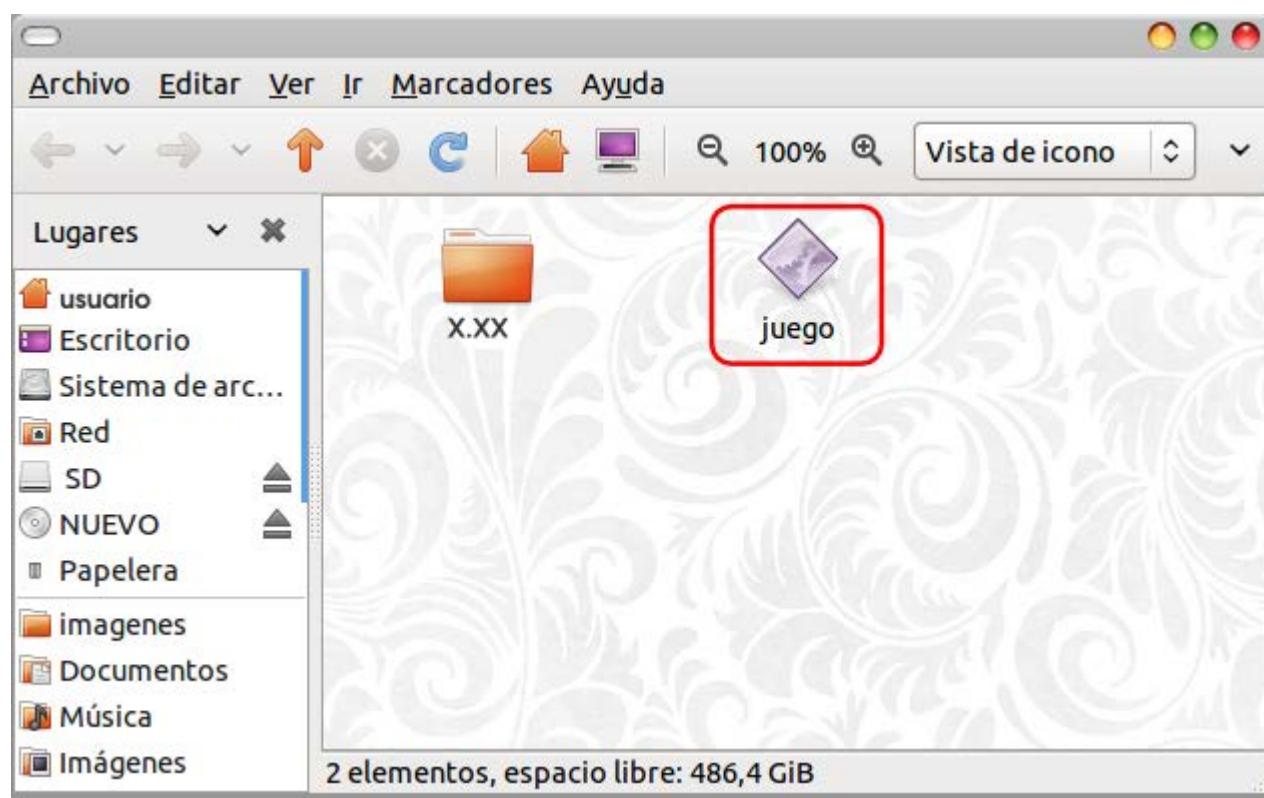
Ya en el entorno de trabajo encontramos la opción en **Archivo/Exportar/Save As Game Engine Runtime**.



Le damos un nombre y lugar de destino. Ya tenemos el autoejecutable. En el caso de Windows se crea un buen repertorio de librerías .DLL. En este ejemplo para Windows nuestra simulación se llama *juego.exe*.

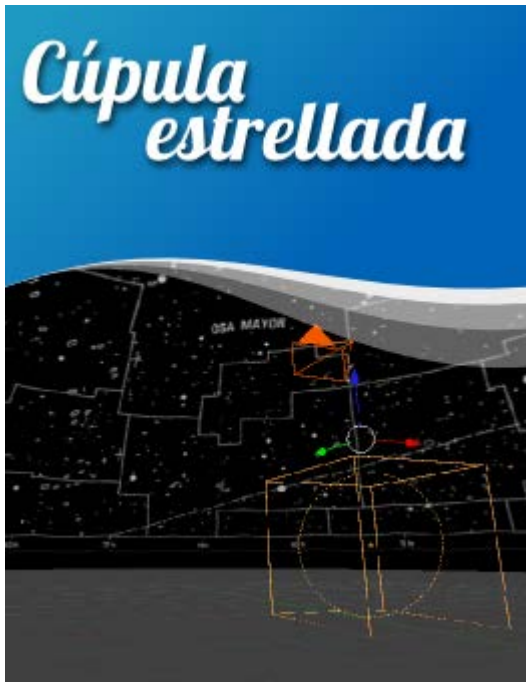


En Linux y MacOSX el producto final es más simple al no tener que añadir librerías.



Exclusividad

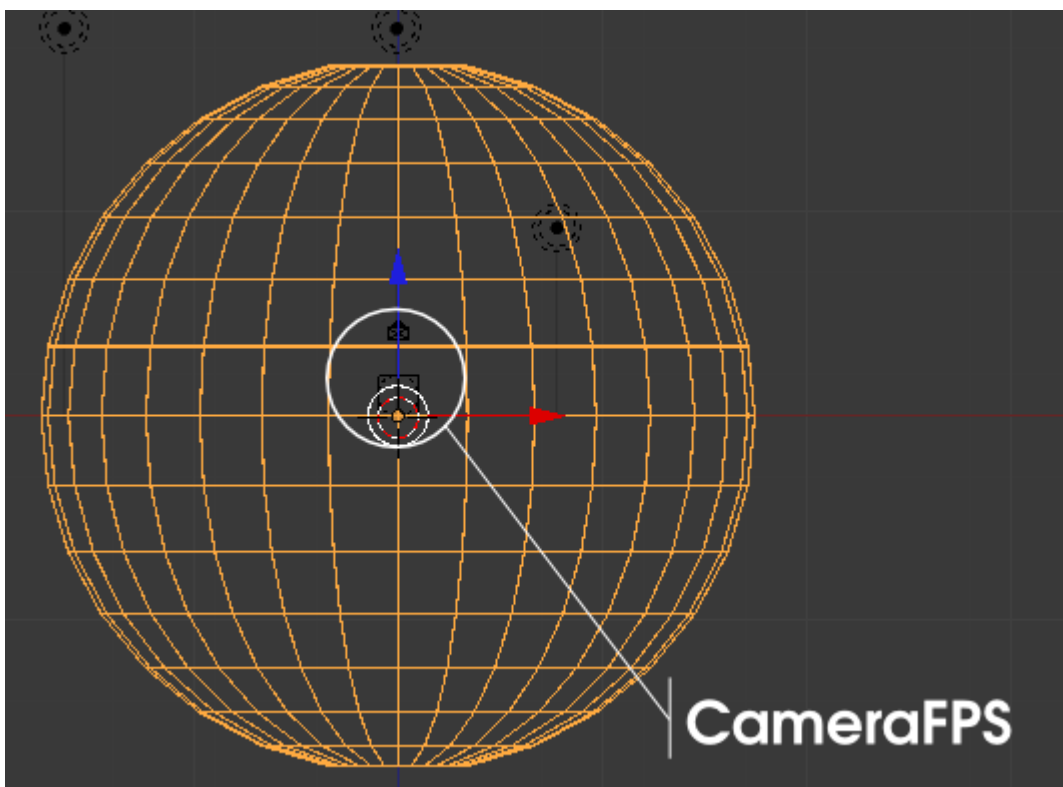
El autoejecutable sólo funciona en la plataforma en la que ha sido creado. Es decir; que un autoejecutable fabricado en un Blender bajo Windows no funciona ni en un sistema operativo Linux ni en uno MacOSX de la misma manera que un Blender compilado para un sistema no puede instalarse en otro.



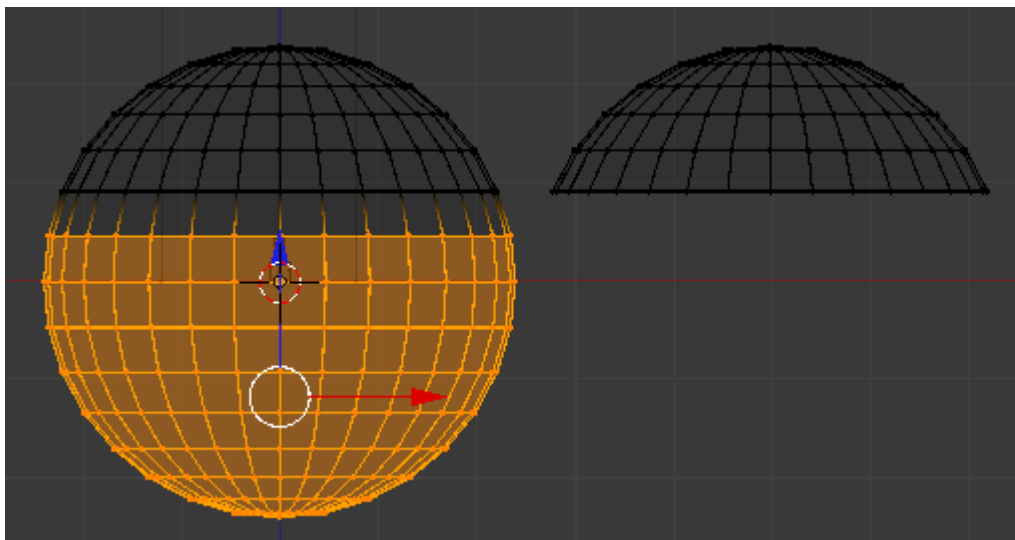
Vamos a crear una **cúpula celeste** con todo el cúmulo de **estrellas del hemisferio norte** para pasear por su interior y mirar al punto que más nos interese en cada momento.

La escena es realmente muy sencilla. Para comenzar:


- Abrimos el archivo *cameraFPS.blend* y eliminamos el plano del suelo. No continuamos sin hacer **Archivo/Guardar como** para no perder el archivo original. Lo llamamos, por ejemplo, *cupula_estrellada.blend*.
- Desde el cuadro **Propiedades ("N")** nos aseguramos de que el **Cursor 3D** se encuentra en las coordenadas **0.0.0**.
- Sacamos una esfera (**Añadir/Malla/Esfera UV**), la **escalamos ("S")** de acuerdo a la escala de **CameraFPS**.



- Le eliminamos los vértices que se indican en esta imagen.



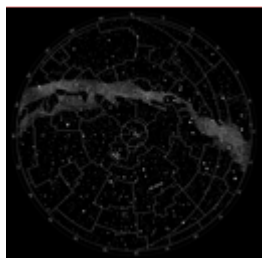
- Le asignamos un **Material**  simplemente pulsando **Nuevo** y le activamos **Textura en caras**.

Hay que mapear la cúpula desde el interior. Invertimos la dirección de las **Normales** porque una esfera las trae mirando hacia el exterior. Ya sabemos que para este tipo de ediciones es muy conveniente tenerlas a la vista (en **Modo Edición** , en la botonera **Visualización de malla** del cuadro **Propiedades "N"**). Seleccionamos todos los vértices ("**A**") y usamos **Malla/Normales/Voltrear normales**.

Para el mapeado necesitamos esta imagen.




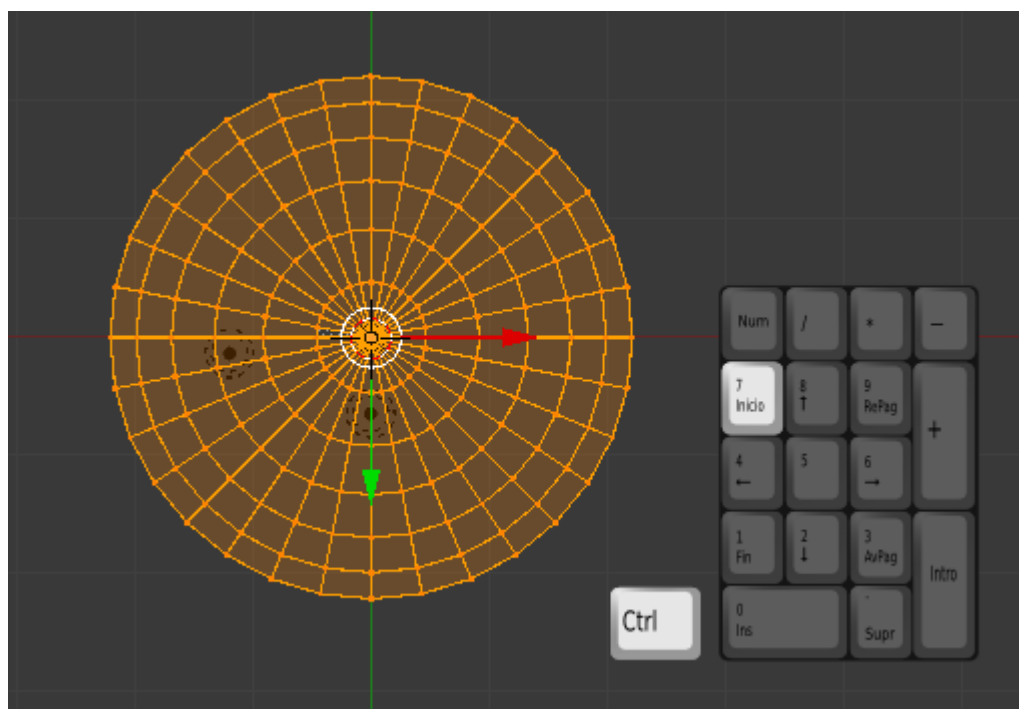
Carta celeste hemisferio norte



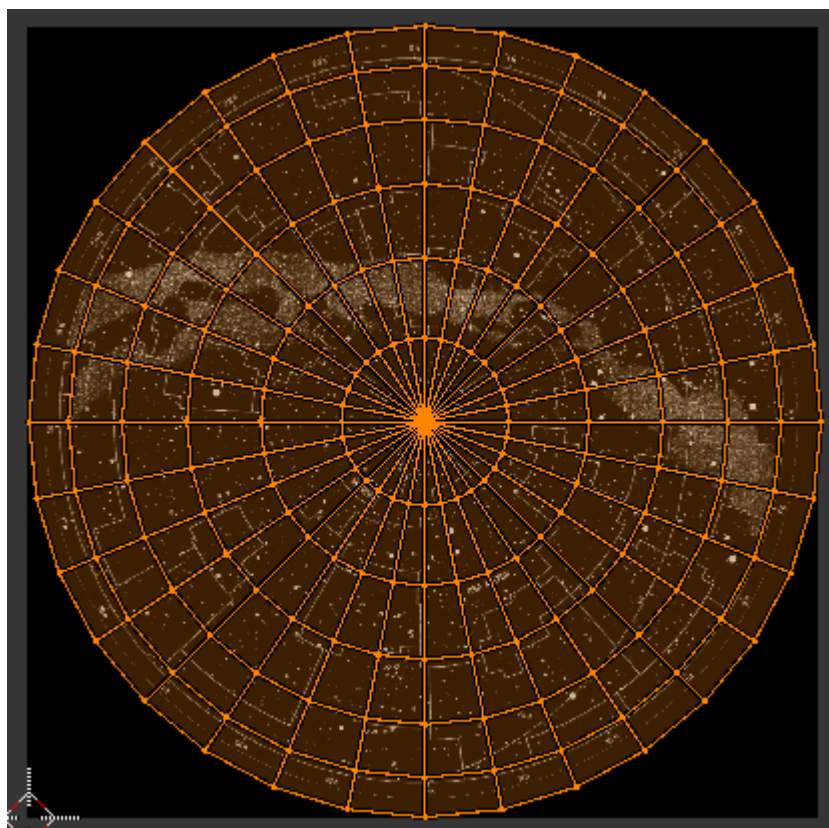
Carta celeste áreas
(obra derivada) // Autor:
Shadowxfox // Licencia:
GNU Free
Documentation License

Nos vamos al entorno de trabajo **UV Editing**. Allí:


- En el **Editor UV**  cargamos la imagen que ya tenemos en el disco duro.
- En el editor **Vista 3D** seleccionamos todas las caras ("**A**") de la cúpula y nos colocamos en el punto de vista en planta pero desde abajo ("**Control_Numpad 7**")

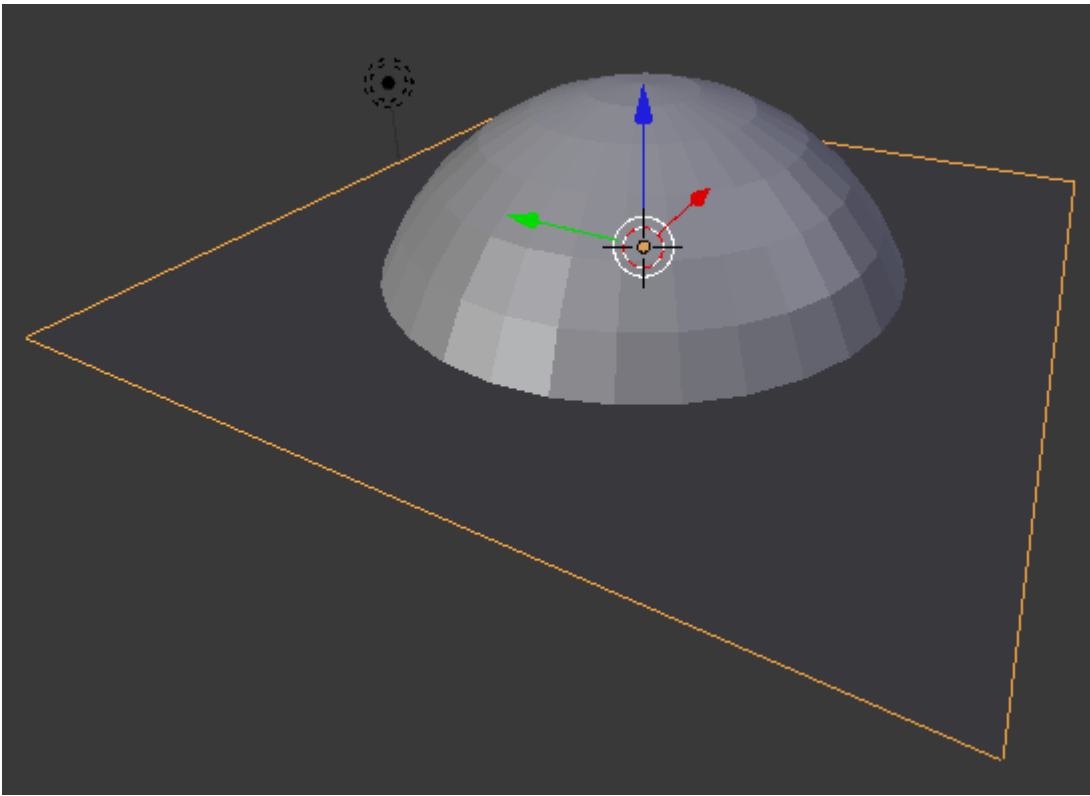


- También en el editor **Vista 3D** ordenamos **Malla/Desplegar UV/Proyectar desde vista (Límites)**. La opción que incluye **Límites** es interesante cuando la imagen está correctamente proporcionada, como es este caso.



De vuelta al entorno de trabajo **Default** solo nos queda:

- Desde el cuadro **Propiedades ("N")** asegurarnos de que el **Cursor 3D** se encuentra en las coordenadas **0.0.0**.
- Añadir un plano (**Añadir/Malla/Plano**) y lo **escalarlo ("S")**. Nos aseguramos de que la **Normal** apunta hacia arriba, le asignamos un **Material**  y le activamos **Textura en caras**.
- Descender la cúpula para que quede apoyada en el plano.



Algunos consejos

- Al menos una de las lámparas debe estar dentro de la cúpula aunque para este paseo virtual tan sencillo lo mejor es dejar sólo una y calcular una buena **Energía** para ella.
- El material de la cúpula es mejor que tenga una **Intensidad: 0.000** para **Especular**. No tiene sentido que se genere una sensación volumétrica de ese tipo en el interior.
- Le aplicamos un sombreado **Suave** a la cúpula. **No es necesario** aplicar modificador **Subdivisión**.

No olvidaremos poner un sombreado **Textura**. Este es el aspecto en la simulación ("P").



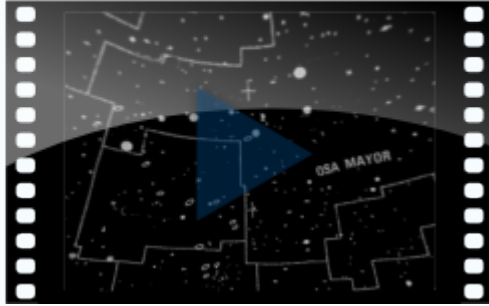
Nada espectacular al tratarse de una imagen fija; el verdadero potencial está en pasear y mirar por el interior de la cúpula.



Vídeo-demostración de la simulación



Vídeo Cupula estrellada



Analiza y estudia el archivo .blend

Usa este .blend para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.



Archivo
cupula_estrellada.blend

Actividades



1- Juega con una vidriera

Con lo que sabes sobre mapeado de texturas con transparencia haz todo tipo de pruebas usando esta imagen de una vidriera de la *Catedral de León*.

Se trata de un PNG con fondo transparente.



Vidriera Catedral de
León (obra derivada) //
Autor: Rodelar //
Licencia : GNU Free
Documentation License




2- Controla el "Cuerpo rígido"

Pasa un rato alterando las opciones de un **Cuerpo rígido** y haciendo pruebas para comprobar cómo la diferencia entre un comportamiento creíble y otro estrambótico puede estar en un solo parámetro.



3- Animación slow-motion

Haz una animación de una simulación con **CPS** (en las físicas del **Mundo** ) a **24** y repítela a **100**. La de **100** se comportará como una película de tipo *slow-motion* (cámara super-lenta).



4- Bloques lógicos de CameraFPS

Una vez que has visto un poco sobre bloques lógicos en **Material didáctico: Galileo en Pisa**, anímate a ver los bloques de **CameraFPS**. Si te animas a tocar algo no olvides hacer antes una copia de seguridad.

Un reto: cambia los controles de dirección del cubo del artilugio para que respondan a los **cursores** en lugar de a **"W"**, **"S"**, **"A"** y **"D"**.



5- Otras proyecciones

Aunque no hayamos profundizado en ello echa un vistazo a los diferentes modos de proyección para el mapeado UV. Practica sobre todo **Proyectar desde vista**, que es uno de los más útiles para nuestros fines.



6- GLSL en el museo

Intenta conseguir la sombra de un cuadro en la pared del museo con el que hemos trabajado.

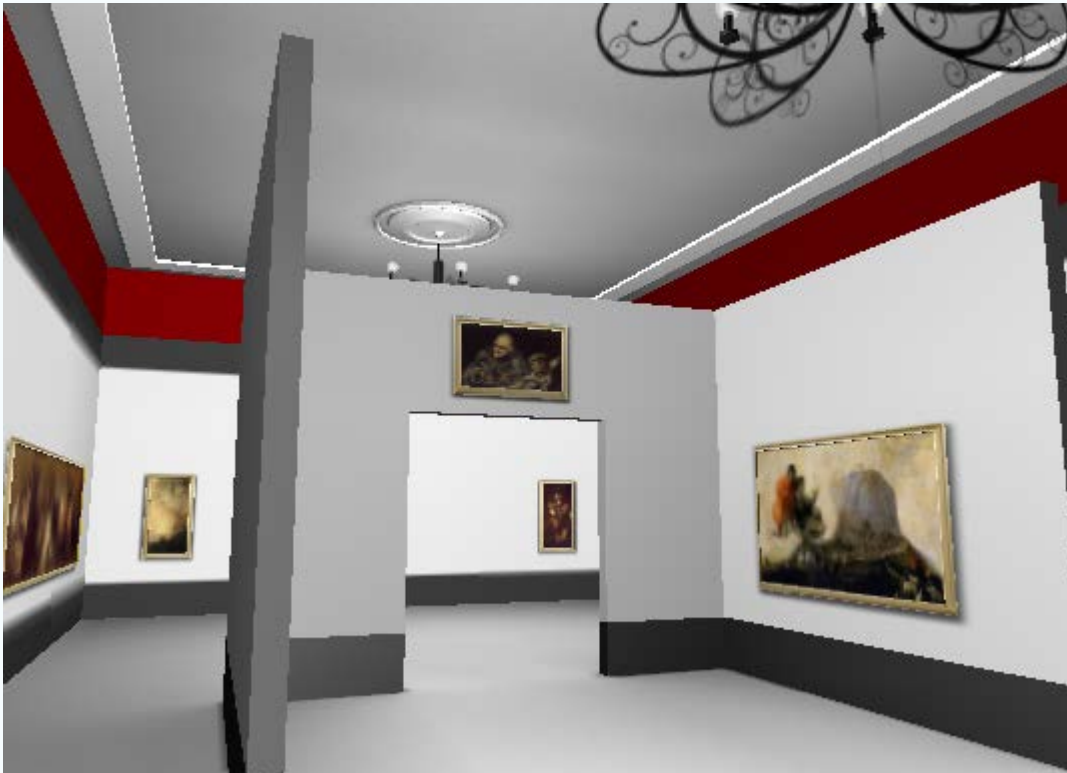


7- Suelo y techo simulados

Realiza las texturas simuladas para el suelo y el techo del museo.



8- Culmina el museo



Al proyecto de pinacoteca le faltan cuadros. Usa las siguientes imágenes y acábalo.

Analiza y estudia el archivo .blend (úsalo para compararlo con tu resultado una vez que hayas realizado toda la práctica. Te servirá de referencia para autoevaluarte.)



La lectura // Autor: Francisco de Goya // Licencia: Dominio público



Dos mujeres y un hombre // Autor: Francisco de Goya // Licencia: Dominio público



Peregrinación a la fuente de San Isidro (el santo oficio) // Autor: Francisco de Goya // Licencia: Dominio público



Duelo a garrotazos // Autor: Francisco de Goya // Licencia: Dominio público



Al aquelarre (Asmodea) // Autor: Francisco de Goya // Licencia: Dominio público



Una manola: doña Leocadia Zorrilla // Autor: Francisco de Goya // Licencia: Dominio público



Perro semihundido // Autor: Francisco de Goya // Licencia: Dominio público



Judith y Holofernes // Autor: Francisco de Goya // Licencia: Dominio público

público



Saturno devorando a un hijo//
 Autor: Francisco de Goya //
 Licencia: Dominio público



Romería de San Isidro // Autor:
 Francisco de Goya // Licencia:
 Dominio público



Aquelarre // Autor: Francisco de
 Goya // Licencia: Dominio
 público

Test de autoevaluación



Autoevaluación: Simulaciones físicas y paseos virtuales

1- El motor de físicas integrado en Blender se llama...

- Ogre
- Sumo
- Bullet

2- Si algunos objetos se comportan de modo extraño, atravesando paredes, en la simulación...

- Aumentamos el valor de Sub-pasos.
- Disminuimos el valor de CPS.
- Disminuimos el valor de Gravedad.

3- Un objetos no regular debe tener un "Limite de colisión"...

- Capsula
- Envoltura convexa
- Desactivado

4- Ponemos en marcha una simulación con...

- "P" en Modo Objeto.
- "P" en Modo Edición.
- "P" en Modo Objeto o en Modo Edición; es indistinto.

5.- El sombreado GLSL solo consigue sombras de lámparas...

- Puntual
- Foco
- Sol

6- El sombreado con el que vemos la simulación en condiciones correctas es...

- Textura
- Sólido
- Alambre

7- Para que un "Cuerpo rígido" rebote...

No hay que hacer nada, por ser "Cuerpo rígido" rebota sin más.

Hay que editar el material en la botonera Físicas.

Hay que aumentar su Radio.

8- Un "Cuerpo rígido" con Amortiguación en Traslación de valor 1.000...

No rebota al colisionar.

No cae.

Sube hacia arriba en contra de la fuerza de la gravedad.

9- Si un objeto tiene material (y siempre debe tenerlo en una simulación), ¿qué opción debe estar activada para que se vean las texturas mapeadas?

Textura en caras.

Desechar caras traseras.

Muestreo completo

10- ¿A que se refiere FPS en configuraciones como CameraFPS?

Son las iniciales del programador inicial del *mouselook*.

Frames Per Second (Fotogramas Por Segundo).

First Person Shooter.