

# Blender 3D en la Educación

## Blender 3D en la Educación

**Módulo 8: Interactividad y animaciones directas**

# Interactividad y animaciones directas

Una de las capacidades más apasionantes de Blender es el llamado **Motor de juegos (Blender Game)** para todo tipo de **virtualizaciones, paseos arquitectónicos, recreaciones...** Sin embargo en este módulo no vamos a llegar a ese **Blender Game** sino que nos quedamos en algo más modesto pero igualmente fascinante: la creación de un objeto con el que podremos **interactuar** de un modo sencillo como es rotarlo a nuestra voluntad.

En **Blender: 3D en la Educación** consideramos **animaciones directas** aquellas que permiten generar movimiento sin necesidad de añadir manualmente fotogramas clave, manipular curvas de interpolación... y todo ese repertorio de conceptos a los que el aprendiz debe llegar sólo cuando ya ha asimilado otros **asuntos básicos respecto a la animación 3D**.

## 3DNP

**3DNP** o **3D-No-Plugins** (3D-Sin-Extensiones) publicado por Thorsten Schlüter con licencia GPL es un *.blend* con toda la configuración para realizar una virtualización de un modelo que después se ve desde un navegador web (Firefox, Chrome, Internet Explorer...) con la particularidad de ser interactivo; es decir, que somos nosotros, con el puntero del ratón, los que controlamos la rotación del objeto en la pantalla.

Como nuestra finalidad en **Blender: 3D en la Educación** no es el lenguaje HTML vamos a facilitar todo el código necesario para despreocuparnos y atender exclusivamente al trabajo con Blender.

## Descarga y contenido del ZIP

- Descarga desde la web del autor: [www.thoro.de/page/3dnp-introduction-en](http://www.thoro.de/page/3dnp-introduction-en)
- Descarga directa desde [aquí](#).

Una vez descargado el ZIP con todo lo relativo a 3DNP dentro hay cinco archivos (echamos un vistazo para una mejor comprensión, pero de todo lo que ahí aparece nosotros sólo manipularemos dos archivos):

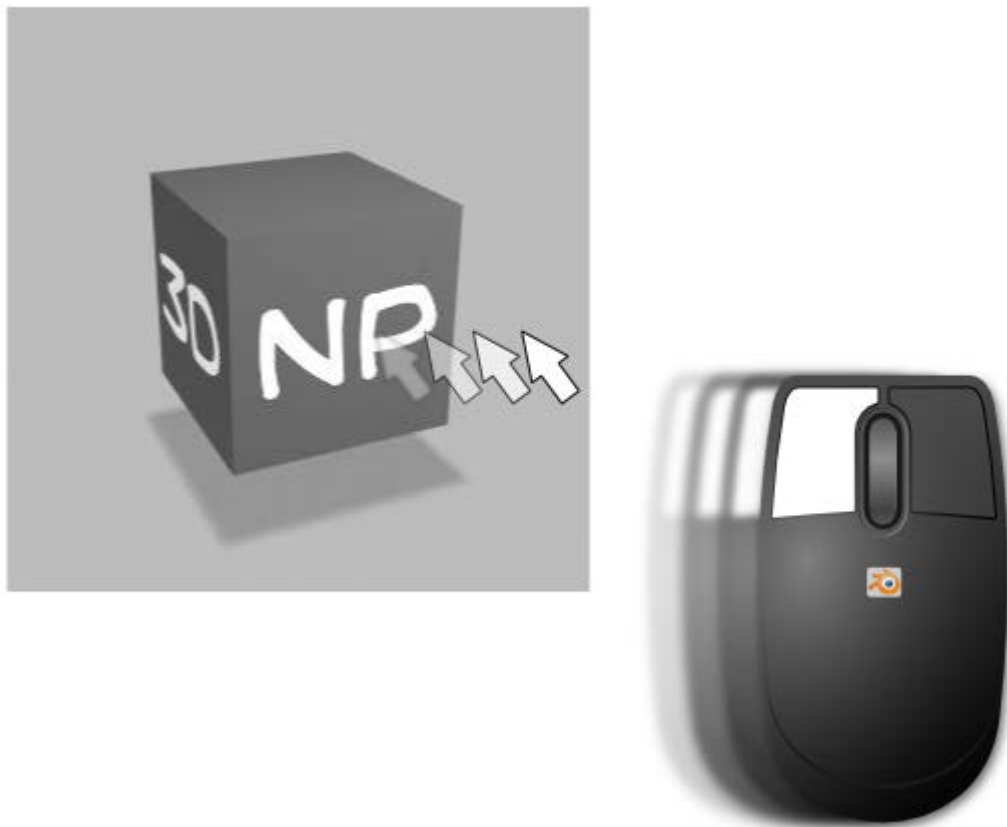
- Carpeta *Blender*. Ahí está el *.blend* que crea todo lo necesario para disfrutar de 3DNP.
- Carpeta *HTML*. Incluye el archivo javascript llamado *3DNP\_config.js*, un archivo CSS con las hojas de estilo de los HTML, unas imágenes y un ejemplo de cada una de las tres formas que tenemos de ejecutar 3DNP desde el entorno web:
  - *3DNP.html*. Carga directamente las imágenes y deja lista la escena para ser rotada (de hecho la imagen comienza a rotar sola). El inconveniente es que el usuario no es avisado de que las imágenes se están cargando y puede frustrar el intento de navegación.
  - *3DNP\_click.html*. Muestra el logo de 3DNP estático, y un enlace "click here to load", que es necesario ser activado por el navegante. Una vez cargada la escena se pone en movimiento a la espera de la navegación del usuario.
  - *3DNP\_loader.html*. Igual que la anterior pero no necesita ser ejecutada por el navegante. Se aprecia la barra de carga por lo que el usuario sabe si la escena está en condiciones de ser navegada o no. Igual que las anteriores, una vez hecha la carga la secuencia se pone en movimiento a la espera de que el visitante comience la navegación.
- Logo de 3DNP.
- Condiciones de la licencia.
- PDF con un tutorial en inglés.



**Importante**

La carpeta principal descomprimida, con todas las subcarpetas y archivos, **debe permanecer tal y como la hemos descargado**. Es muy importante que se mantenga toda la estructura para que las rutas relativas entre archivos no se rompan y deje de funcionar 3DNP.

Si ejecutamos el archivo **3DNP\_html** vemos el ejemplo que incorpora de fábrica e interactuamos con él **pulsando el ratón y desplazando**. Tiene algunas características como la rotación inicial que nosotros anularemos a su debido tiempo.



## Entorno de trabajo

Ejecutamos el archivo *3DNP Blender.blend* que se encuentra dentro de la carpeta *Blender*.

Vamos a analizar el entorno de trabajo para saber el terreno que pisamos:

- Abajo del todo hay un nuevo **Editor de texto** con un código en el lugar donde habitualmente encontramos el editor con la **Línea de tiempo**. Ese es el *script* escrito en Python que hace que se genere lo necesario para que funcione 3DNP. Nosotros **no** vamos a manipularlo.

```

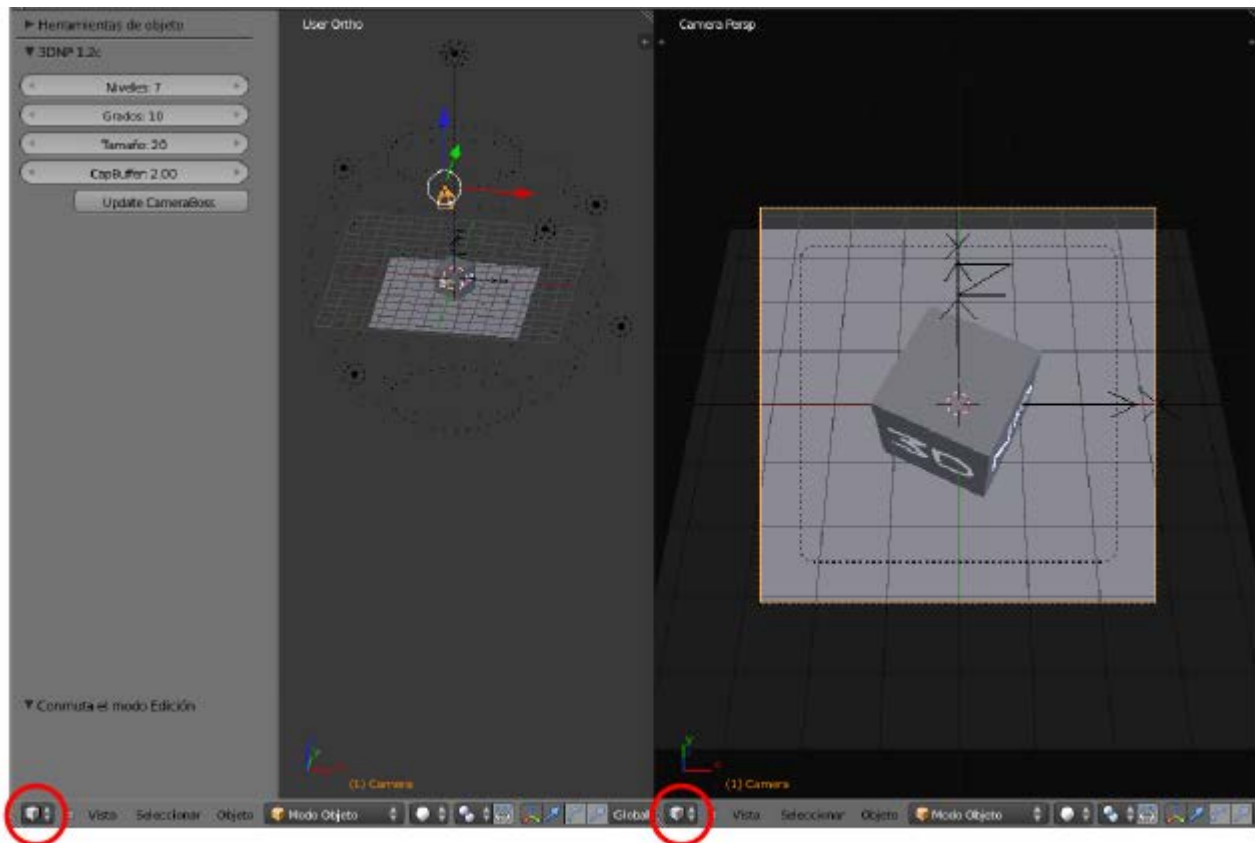
1 #
2 # 3DNP for Blender 2.6
3 # 2011 Thorsten Schlüter www.thoro.de
4 #
5
6 import bpy
7 from bpy.props import *
8 from math import *
9
10 print ("##### 3DNP #####")

```

Vista Texto Editor Formato Plantillas 3DNP.py

- Arriba hay dos editores **Vista 3D**. El de la derecha es para tener un punto de vista desde la cámara y el de la izquierda para tener una vista del artilugio. Por supuesto que podríamos prescindir de uno de ellos, y alternan la vista general con la de la

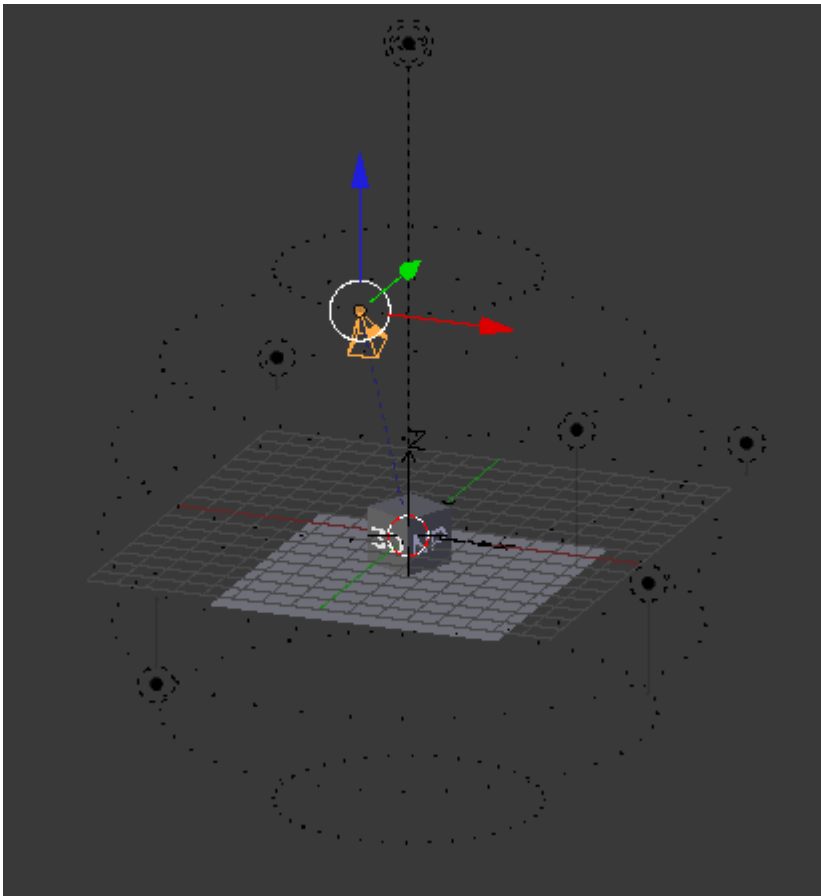
cámara, pero ya que están ahí no le damos mayor importancia y las dejamos estar.



- En el editor **Vista 3D** de la izquierda el cuadro **Herramientas** ("T") cuenta con una novedad. El *script* está ejecutado y hace que se muestren opciones relativas a 3DNP.



- Y en ese editor **Vista 3D** aparece el artillugio con el que trabajar.

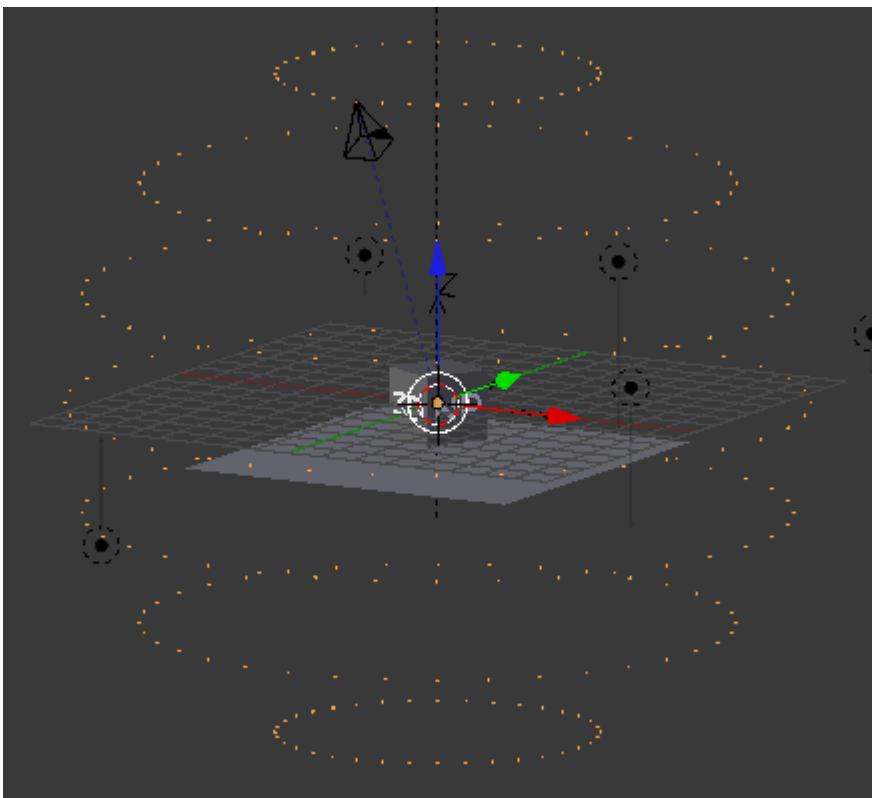


## El artilugio

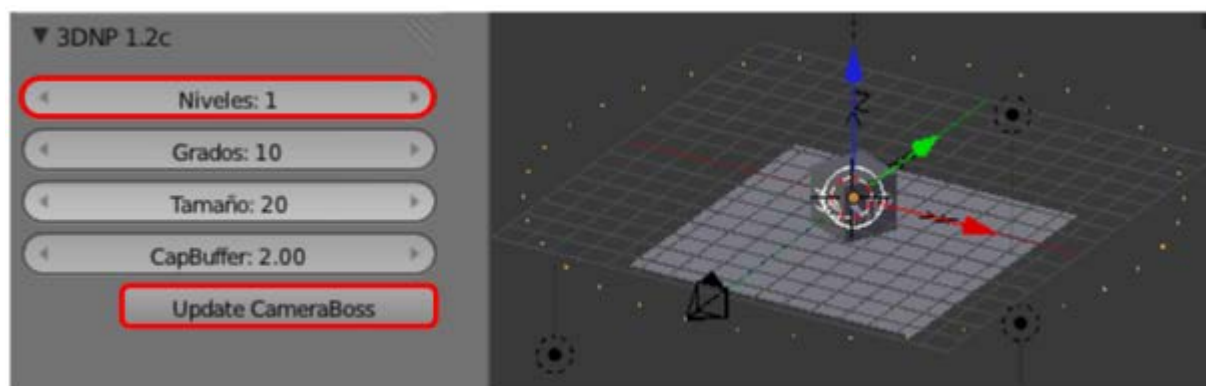
¿Qué es todo eso que aparece en la escena por defecto?. Vamos poco a poco.

Lo que hace 3DNP es obligar a la cámara a desplazarse haciendo círculos alrededor de un objeto mientras crea los *renders* correspondientes y así fabricar una ilusión del movimiento.

Si nos fijamos hay una especie de esfera formada por unos vértices. Ese objeto se puede seleccionar como cualquier otro objeto en Blender.



Los distintos puntos marcan las posiciones que irá tomando la cámara en su recorrido haciendo *renders*. Es lo que en 3DNP se llaman **Niveles** y que nosotros aquí vamos a limitar a **1** (es necesario refrescar pulsando el botón **Update CameraBoss**).



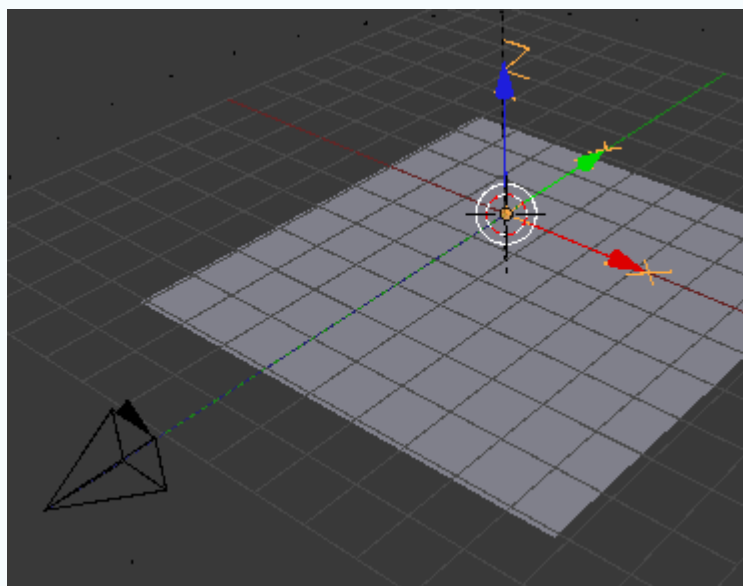
No vamos a cambiar nada más pero conviene tener presente:

- **Grados**. Ángulo entre posición y posición en la circunferencia de cada **Nivel**.
- **Tamaño**. Se refiere a la esfera que delimita la trayectoria de la cámara
- **CapBuffer**. Distancia entre el primer y el último **Nivel**. Esto no afecta al tamaño de la esfera pero ajusta automáticamente la distancia entre todos los **Niveles**.



## La cámara

La configuración de la cámara no es muy especial. Lo único interesante es que si eliminamos el cubo y los textos que vienen de ejemplo se hace visible un objeto **Vacío**.



La cámara está conectada a ese **Vacío** con una línea de puntos que representa la relación que tiene con él: esté donde esté la cámara siempre apuntará a ese objeto.

Esto quiere decir que cuando traigamos nuestro objeto desde otro *.blend* deberemos colocarlo teniendo en cuenta ese detalle.

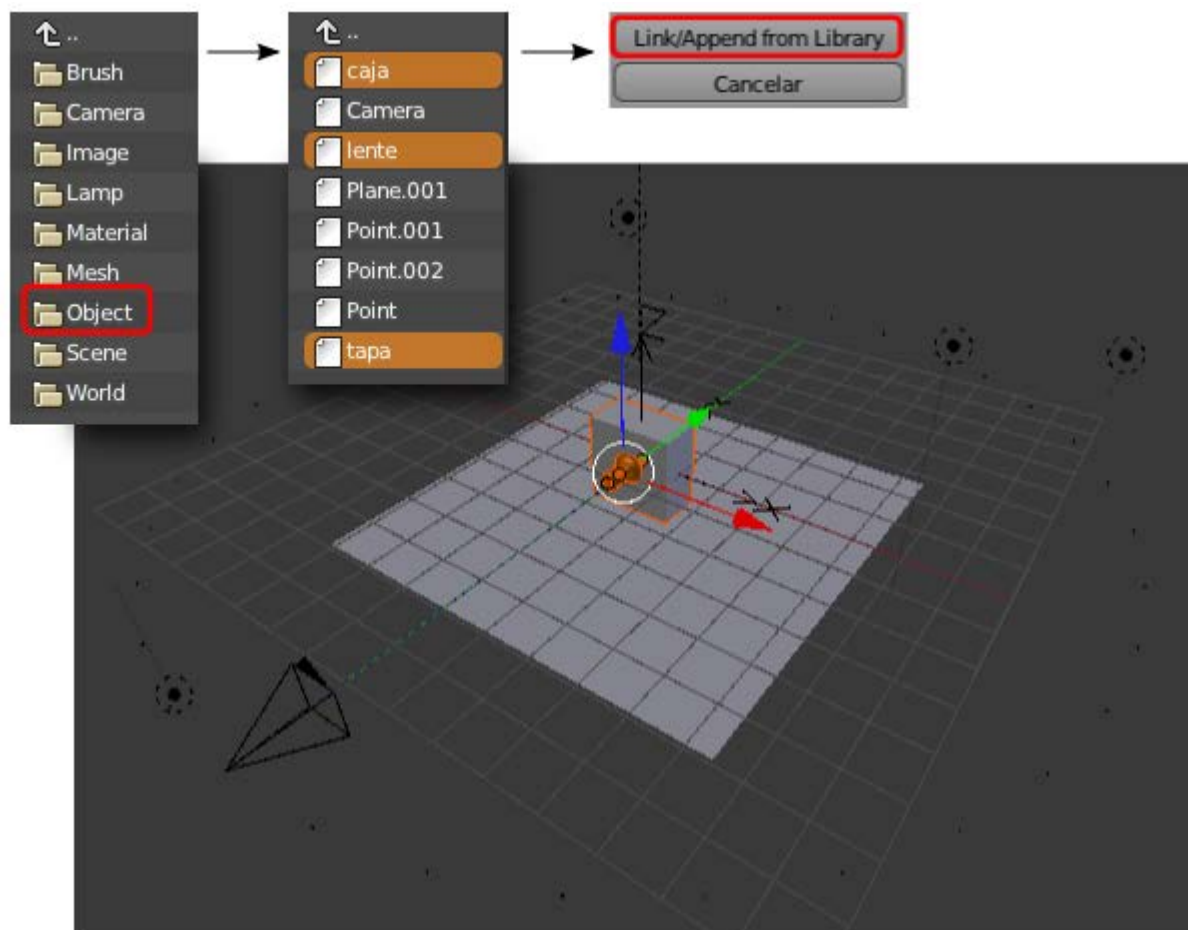
## Importar objetos

En otro *.blend* tenemos el objeto que queremos enviar al archivo con el 3DNP. En nuestro caso se trata de la cámara oscura con una textura mapeada en el cubo del cuerpo.

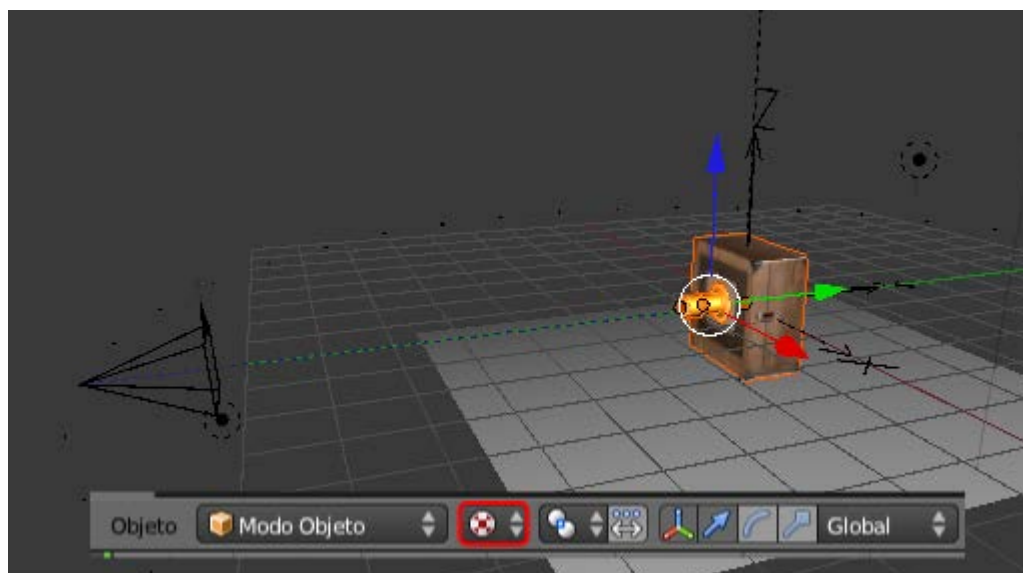


No es necesario abrir ese archivo porque la importación la hacemos desde *3DNP Blender.blend* con el que venimos trabajando. Damos por hecho que se han eliminado el cubo de ejemplo y los textos "3D" y "NP".

Para la importación usamos **Archivo/Añadir** y nos vamos al archivo *.blend* en el que se encuentra la cámara oscura, accedemos a su contenido y entramos en la sección **Object**; una vez dentro escogemos con "**Shift**" (para acumular), los tres objetos que conforman la cámara oscura (en nuestro caso se llaman *caja*, *lente* y *tapa*). Nada más que confirmamos la orden pulsando el botón **Link/Append from Library** estos objetos aparecen en la escena de 3DNP. Además, como en el archivo original el cubo de la cámara está situado en el origen de coordenadas nos podemos despreocupar porque aparece en el sitio adecuado para nuestros fines).



Ya que ese objeto tiene asignada una imagen mapeada debemos activar el modo de sombreado **Textura**.

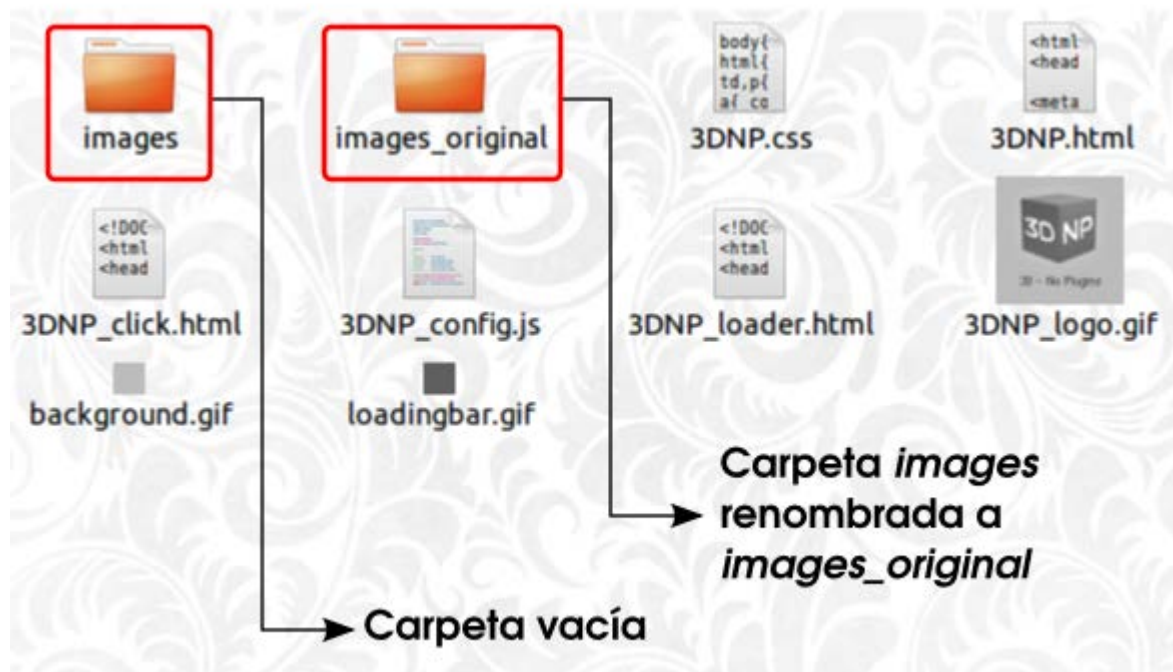


Estamos listos para generar todas las imágenes.

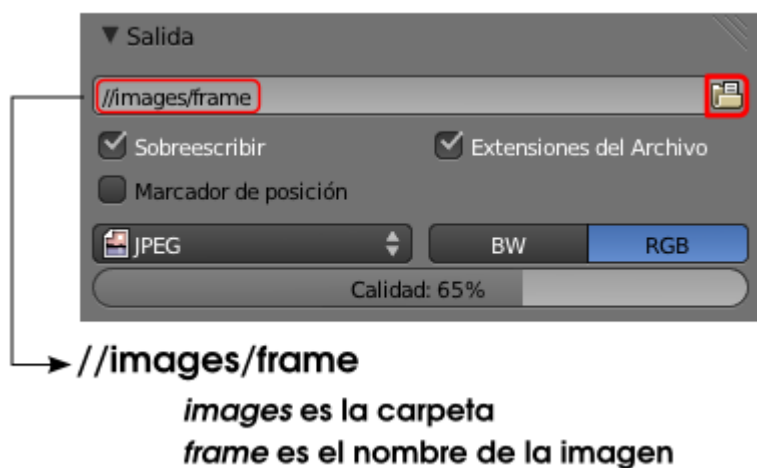


## Exportar las imágenes

Lo primero es preparar el lugar de destino. La carpeta donde acaben todos los *renders* tiene que llamarse *images* y estar al lado de los archivos *3DNP.html*, *3DNP\_click.html* y *3DNP\_loader*. Estos archivos están en la carpeta HTML en la que además vemos que ya hay una carpeta con ese nombre (*images*). Esa carpeta existe porque ya alberga los *renders* del ejemplo que incorpora **3DNP**. Como no tenemos interés en perder ese ejemplo le cambiamos el nombre (de *images* a *images\_original*) y después creamos una carpeta nueva vacía de nombre *images*.




Nos vamos a Blender y en el panel **Render** usamos la botonera **Salida** y el icono de la carpetita para determinar la ruta hacia *images* que acabamos de dejar preparada. Blender le da un nombre automáticamente al archivo de salida; este nombre es *frame* y no lo cambiaremos bajo ningún concepto.

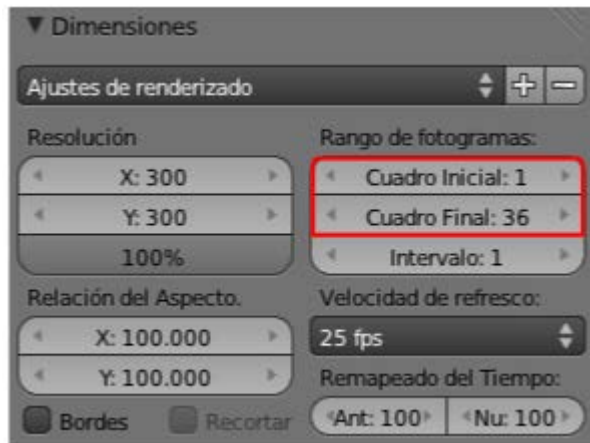


### El peso de las imágenes


Si nos fijamos en la imagen anterior 3DNP exportará en formato JPEG con una calidad del 65%. Este valor tan elevado en la compresión viene determinado por el hecho de que necesitamos imágenes lo más ligeras posibles para facilitar el refresco entre una y otra. Pensemos que es muy habitual poner este material interactivo en web y este tipo de asuntos suelen marcar la diferencia entre una navegación confortable y otra que no lo es.

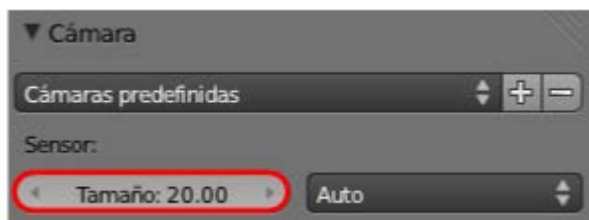



3DNP ha calculado automáticamente el número de *renders* que tiene que realizar al ser **10** los grados entre posición y posición; concretamente necesita **36** (360/10). Blender nos informa de este dato en el panel **Render**  en la botonera **Dimensiones**.



Añadimos un par de cambios más:

- El tamaño de los renders con **Resolución X** y **Resolución Y** a **450**.
- Seleccionamos la cámara y en su panel  cambiamos el valor **Tamaño** a **20.00**.



Y, definitivamente, ha llegado el momento de crear los *renders*. En esta ocasión usaremos el botón **Animación** del panel **Render** .



Tras una dosis de paciencia la carpeta *images* acabará llena con los 36 *renders*.

## La configuración del javascript

Si nos dirigimos a la carpeta *HTML* de 3DNP y ejecutamos el archivo *3DNP.html* el resultado no será el esperado. Es cierto que si hacemos **clic y arrastramos** es posible que llegue a hacerse visible nuestro modelo interactivo. Pero nosotros queremos que quede correcto.

¿Qué ha ocurrido? Pues sencillamente que 3DNP viene configurado para trabajar correctamente si se dejan los valores por efecto con **Niveles: 7**. El cambio de **7** a **1** hace que el código no funcione bien. ¿Qué código?: el javascript que permite la interactividad; es decir el archivo llamado *3DNP config.js* que está dentro de la carpeta *HTML*.

Sería fantástico que al hacer clic en **Update CameraBoss** al trabajar en 3DNP se autoeditara el código javascript, pero no es así. Tenemos que hacerlo a mano. No es nada complicado.

Accedemos al código javascript de ese archivo y para ello basta abrirlo con un editor de textos cualquiera, como puede ser:

- **Gedit** en Linux.
- **Bloc de Notas** en Windows.
- **Textedit** en MacOS.



## El código por defecto

Además de un buen número de líneas con comentarios, este es el código del javascript que contiene ese archivo:

```
total = 144;
levels = 4;
startlevel = 4;

filemode = 'NameNumber';
filename = 'frame';
suffix = '.jpg';
barLength = 164;

viewmode = 'object';
friction = 0.5;
rotomatic = 80;
rotoresume = 3;

keycodes = [119,100,115,97]
```

Esto es lo que hay que alterar:

- **Total.** De **144** a **36**. El valor **144** viene determinado porque hay un plano que hace de suelo y que recoge la sombra. De los siete niveles por defecto sólo nos hubieran interesado los cuatro superiores ( $36 \times 4 = 144$ ); los tres niveles inferiores serían ignorados aunque se hubieran hecho los *renders*. Nosotros hemos descendido los **Niveles** a **1** así que está claro que este parámetro tiene que ser **36**.
- **Levels.** De **4** a **1**. Recordemos que nosotros habíamos cambiado **7** por **1**.
- **Starlevels.** De **4** a **1**. La orden era que la virtualización comenzara en el nivel **4** (el central) con una vista frontal del objeto. Con el cambio de **Niveles** a **1** no tiene sentido decirle al programa que comience en el nivel **4**. Se produce una incoherencia, por lo que hay que cambiarlo a **1** para que todo tenga lógica.
- **Rotomatic.** De **80** a **0**. Es la velocidad y el sentido de la rotación automática. El *script* da la orden de comenzar una rotación automática en el caso de que el navegante permanezca pasivo. Los valores positivos y negativos determinan el sentido. Pero a nosotros nos interesa en valor **0** porque hace que el efecto se desactive.
- **Rotoresume.** De **3** a **0**. Es el tiempo en segundos que el *script* espera para poner en marcha el anterior parámetro **Rotomatic**. Nos vuelve a interesar el valor **0** para anularlo, aunque es cierto que con **Rotomatic** a **0** es intrascendente manipular **Rotoresume**.



## El código definitivo

```
total = 36;
levels = 1;
startlevel = 1;

filemode = 'NameNumber';
filename = 'frame';
suffix = '.jpg';
barLength = 164;

viewmode = 'object';
friction = 0.5;
```

```
rotomatic = 0;  
rotoresume = 0;
```

```
keycodes = [119,100,115,97]
```

## Interactividad en web

No es nuestra intención introducirnos en los lenguajes Python, HTML, CSS, JavaScript... así que vamos a facilitar todo lo necesario para nuestras presentaciones de 3DNP.

Cuando decimos "integración en web" nos referimos exclusivamente a visualizar nuestra virtualización con un navegador como Firefox, Internet Explorer o Safari.

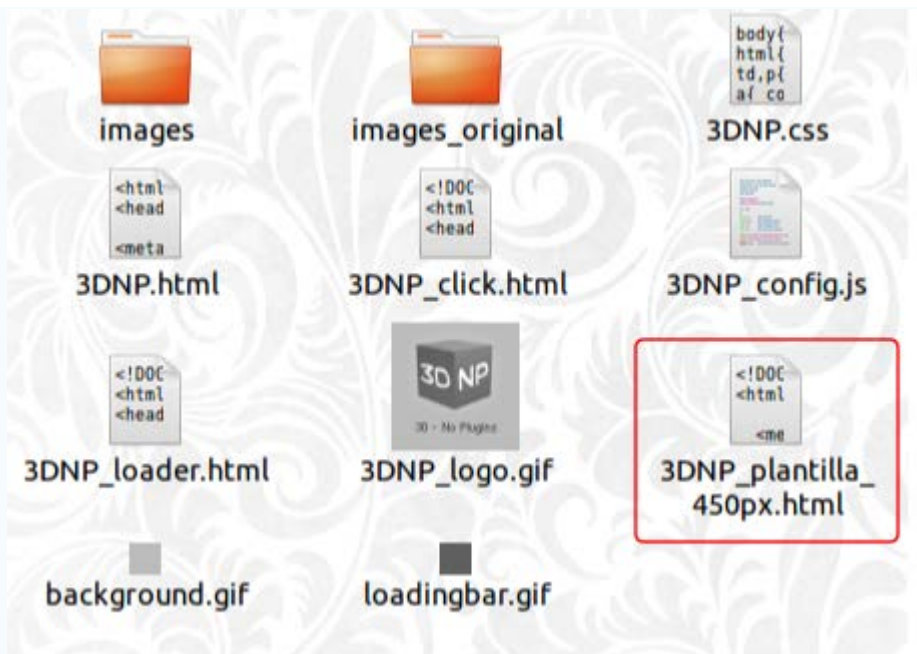
Hemos preparado una plantilla HTML para visualizar en el centro de la pantalla virtualizaciones de 450x450px.



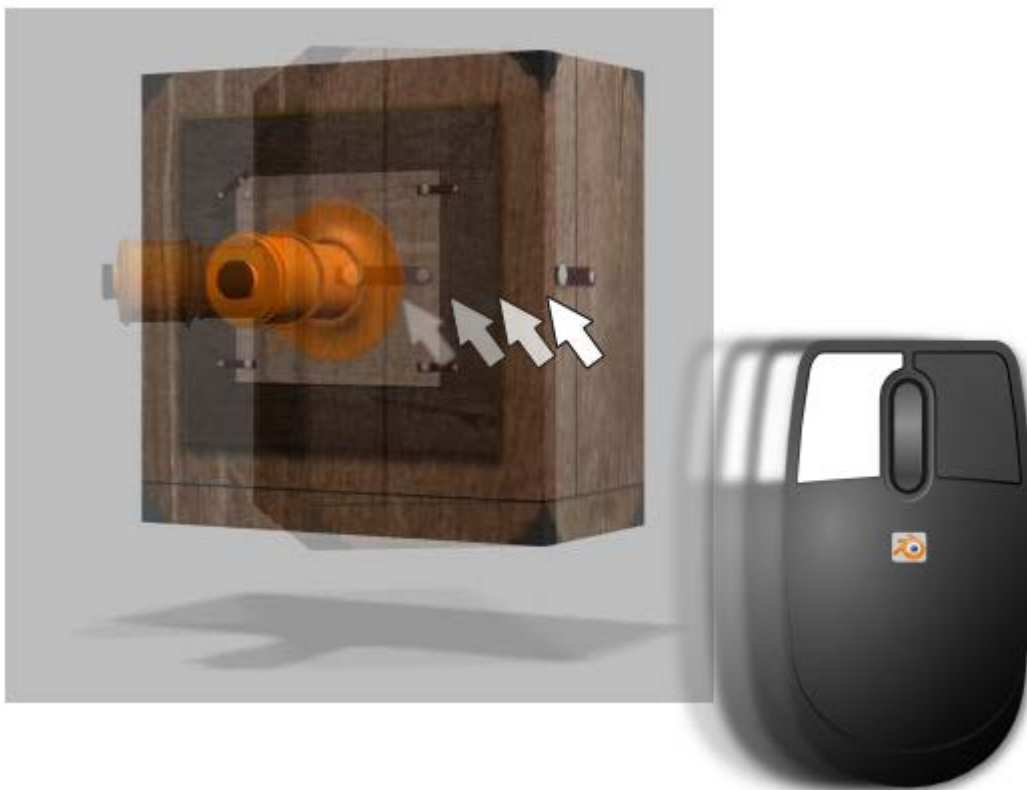
### El código de la plantilla

Sólo hay que **Copiar/pegar** este código en un editor de texto y guardarlo con el siguiente nombre y extensión *3DNP\_plantilla\_450px.html*. El directorio de destino será la carpeta *HTML* de 3DNP.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">  
  
<html>  
  
<head>  
  
<meta content="text/html; charset=ISO-8859-1" http-equiv="content-type">  
  
<title>3DNP_plantilla_450px</title>  
  
<style type="text/css"></style><link rel="stylesheet" href="3DNP.css" type="text/css">  
  
</head>  
  
<body>  
  
<div id="iframe" style="text-align: center; margin-top: 65px;">  
  
<iframe style="border-style: none;" src="3DNP.html" name="3dnp" align="middle" frameborder="yes" height="450"  
scrolling="no" width="450"></iframe>  
  
</div>  
  
</body>  
  
</html>
```



Una vez hecho esto, basta con ejecutar *3DNP\_plantilla\_450px.html* para interactuar con nuestra virtualización. Tal y como hemos dicho debe ejecutarse con un explorador del tipo Firefox, Safari, Internet Explorer...



### 3DNP\_plantilla\_450px

No hace falta tener ningún conocimiento de HTML para poder adaptar la plantilla a cualquier otro tamaño de *renders*. Sólo hay que acceder al código del archivo *3DNP\_plantilla\_450px.html* con un editor de textos como hicimos con el javascript y cambiar los valores de **height** (alto) y **width** (ancho) ajustándolos a los de los *renders* que hayamos hecho.

```
<style type="text/css"></style><link rel="stylesheet"
href="3DNP.css" type="text/css"></head><body>
<div id="iframe" style="text-align: center; margin-top:
65px;"><iframe style="border-style: none;" src="3DNP.html"
name="3dn" align="middle" frameborder="yes" height="450"
scrolling="no" width="450"></iframe>
</div>
</body></html>
```

Con algunos conocimientos de diseño web se puede mejorar su aspecto con imágenes de fondo, título...



### Analiza y estudia el ZIP

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



**Archivo**  
**3DNP\_camara\_oscura.zip**

NOTA: descomprime y ejecuta el archivo *3DNP\_plantilla\_450px.html*

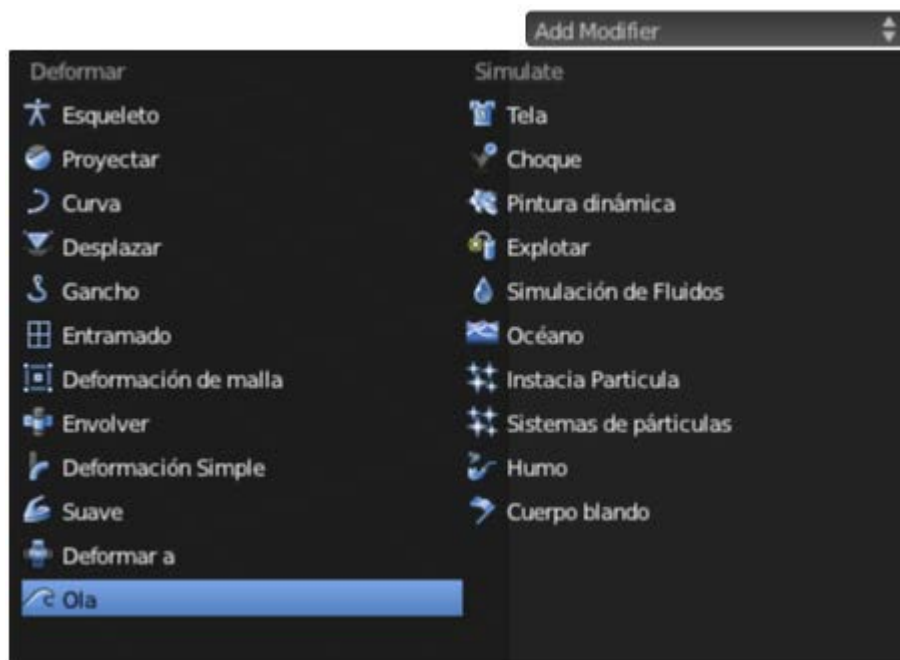
## Animaciones con modificadores

Vamos a considerar **animaciones básicas** aquellas que Blender realiza a través de modificadores y que, por lo tanto, se consiguen mediante la manipulación de sus opciones.

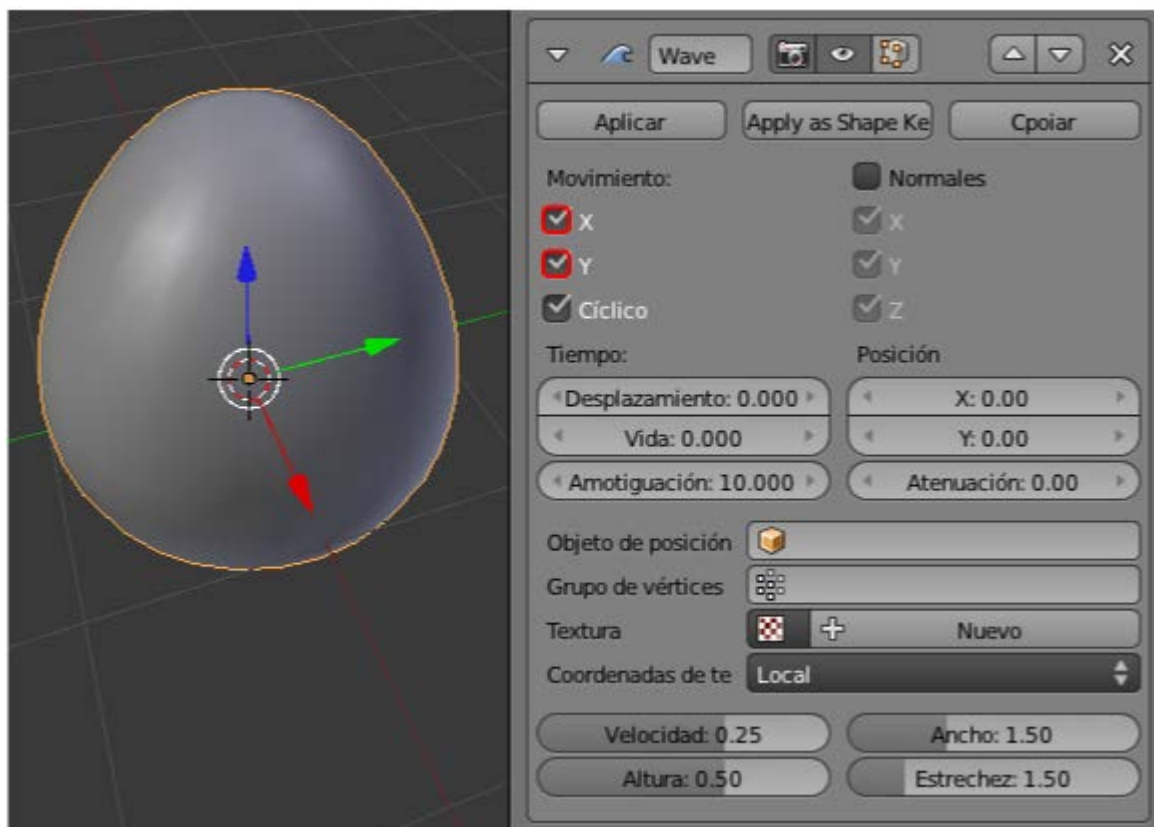
Este tipo de animaciones nos sirven para entrar en contacto con el mundo de la animación familiarizándonos con algunos de sus conceptos más habituales.

### Ola

Supongamos una esfera (**Añadir/Malla/Esfera UV**) a la que le asignamos un modificador **Ola**.



La esfera sufre una deformación inmediata debido a que las opciones **Movimiento X e Y** están activadas.




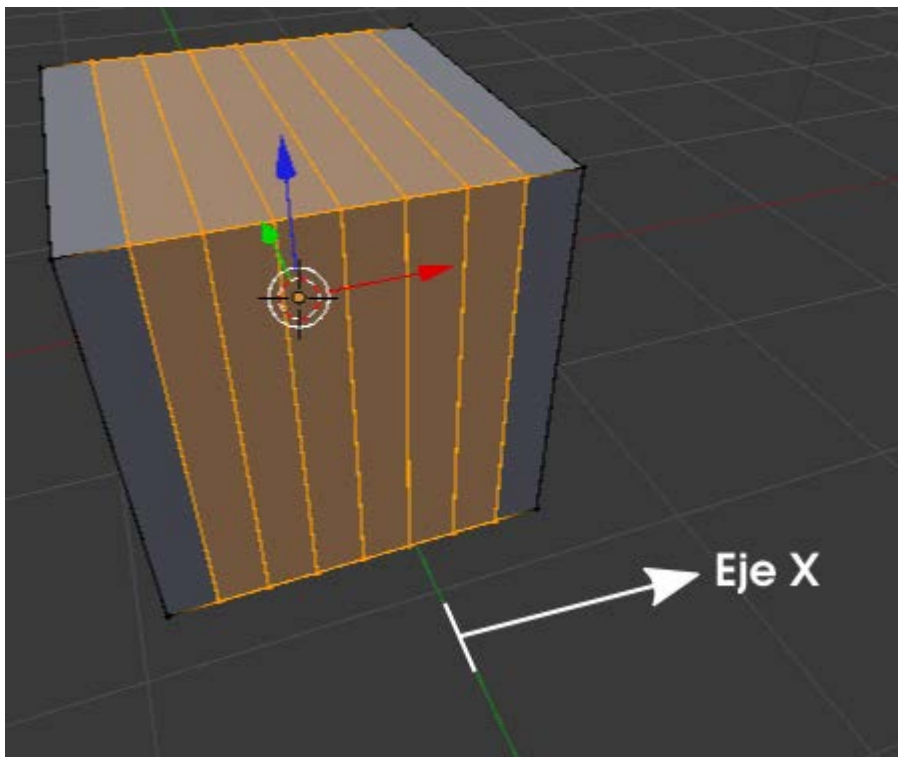
Además de la deformación se ha generado automáticamente una animación asociada al objeto. Para ponerla en marcha en el editor **Vista 3D** usamos el menú **Vista/Reproducir animación** (o el atajo "**Alt\_A**"). Usamos "**Esc**" para salir.




Incluso mientras se reproduce la animación es posible alterar parámetros y opciones del modificador.

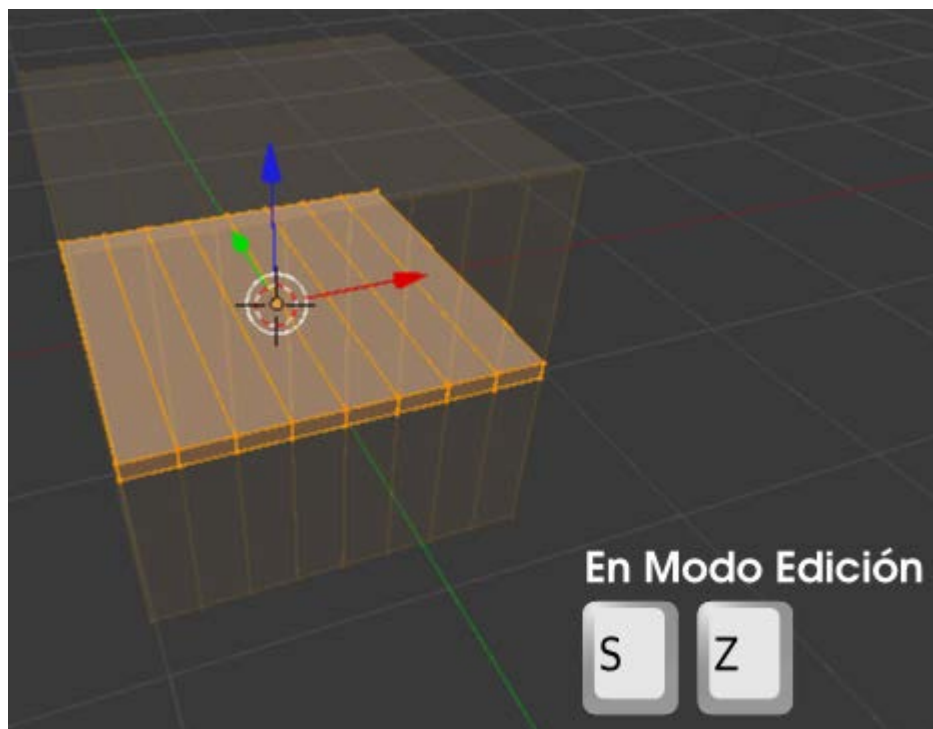
Vamos con una aplicación concreta.


Creamos un cubo (**Agregar/Malla/Cubo**) y en **Modo Edición**  le añadimos varios bucles ("**Control\_R**") alineados con el eje X. Recordamos que para añadir varios bucles a la vez hacemos rodar la rueda del ratón en el momento en el que el bucle virtual está de color fucsia.



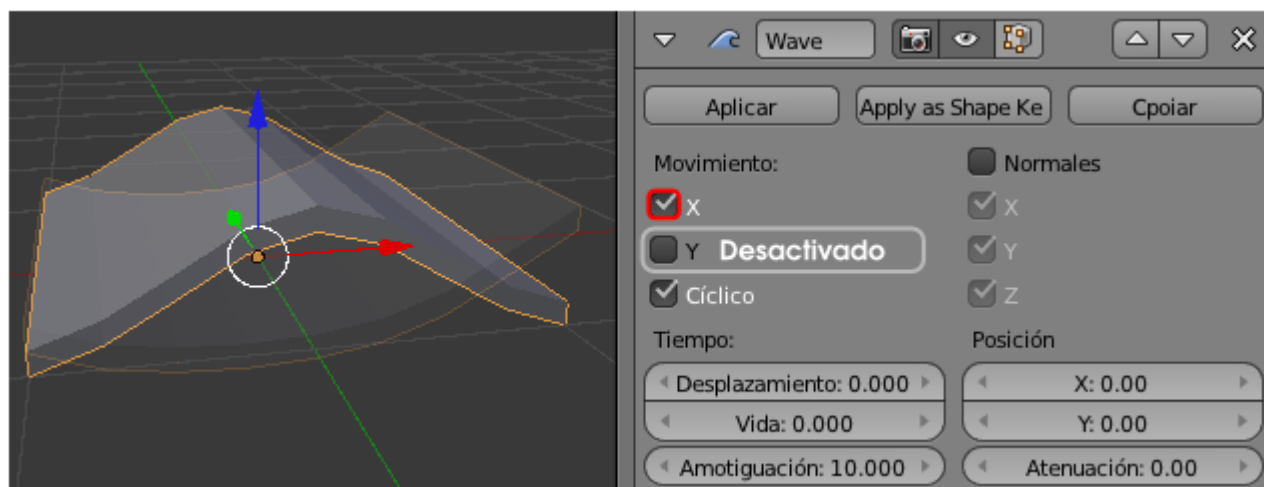
Ahora vamos a escalar en Z para que sea una plancha fina; pero el escalado lo hacemos desde **Modo Edición** , así que seleccionamos todos los vértices ("**A**") y después **escalamos** ("**SZ**").





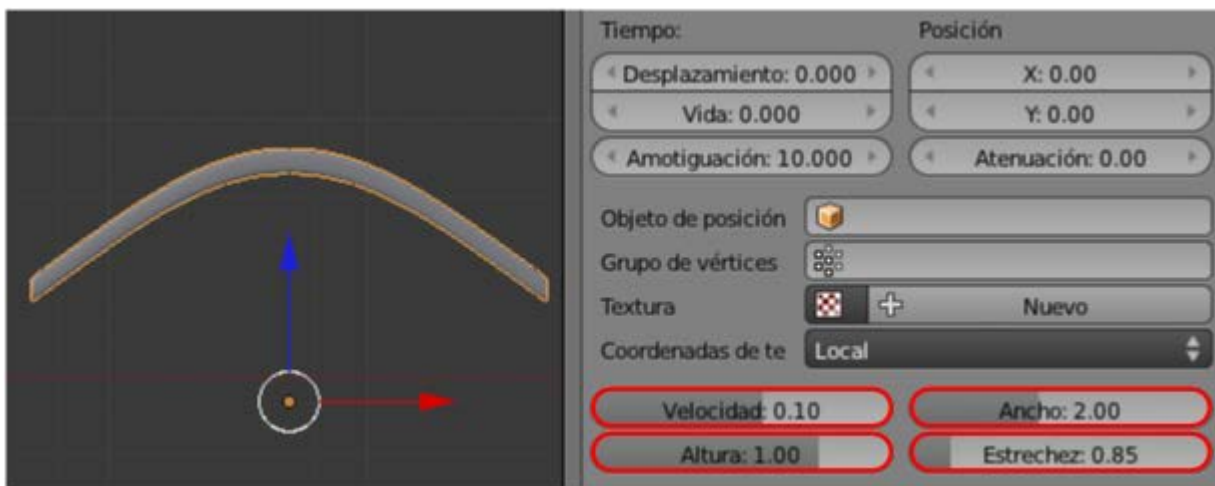
Pasamos a **Modo Objeto**  y le asignamos el modificador **Ola**. Nuestra malla se deforma ligeramente tal y como le pasó antes a la esfera. Nos proponemos representar la longitud de onda del color rojo del espectro. Ni mucho menos queremos una representación exacta, lo que nos interesa es representar en el mismo gráfico la diferencia entre esta y la del color rojo (mucho menos amplia).

Como los nuevos bucles se crearon paralelos al eje X vamos a hacer que las ondas sólo se generen en ese eje, así que desactivamos **Movimiento: Y**.

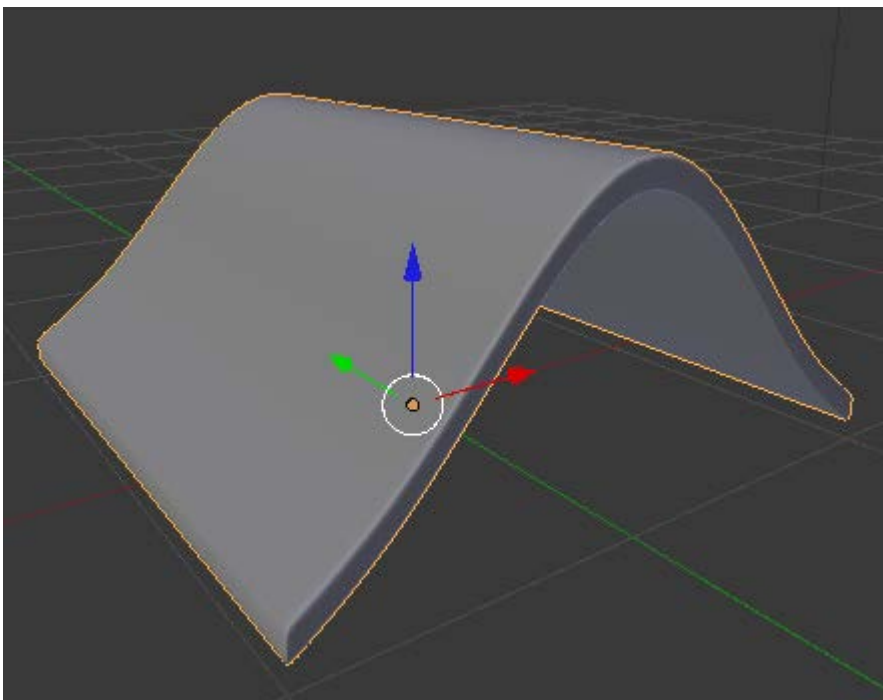


En la parte de abajo de las opciones definimos:

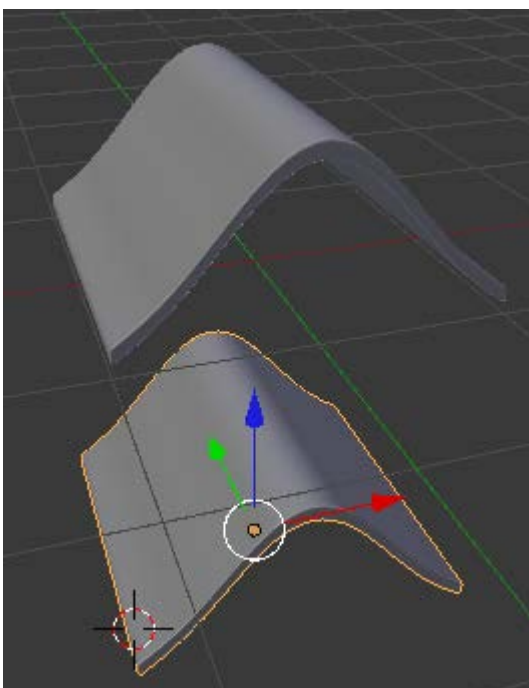
- **Velocidad: 0.10.** Para que vaya lento y se pueda apreciar bien el concepto de la longitud de onda.
- **Altura: 1.00.** Es decir, la altura de las crestas de la longitud de onda.
- **Ancho: 2.00.** La longitud de onda propiamente dicha, es decir, la distancia entre *cresta* y *cresta*.
- **Estrechez: 0.85.** Este parámetro determina la distancia entre la *cresta* y el *valle* de la onda. Con este valor conseguimos que quede muy simétrica. Si no editamos este parámetro es muy probable que la onda se parezca más a un cardiograma que al efecto constante que buscamos.



No estaría mal perfeccionar la malla con un modificador **Subdivisión**, con el sombreado **Suave**, y con unos nuevos bucles en las zonas necesarias presentará este aspecto.



Antes de asignarle material **duplicamos el objeto** ("Shift\_D") y lo desplazamos en Z.

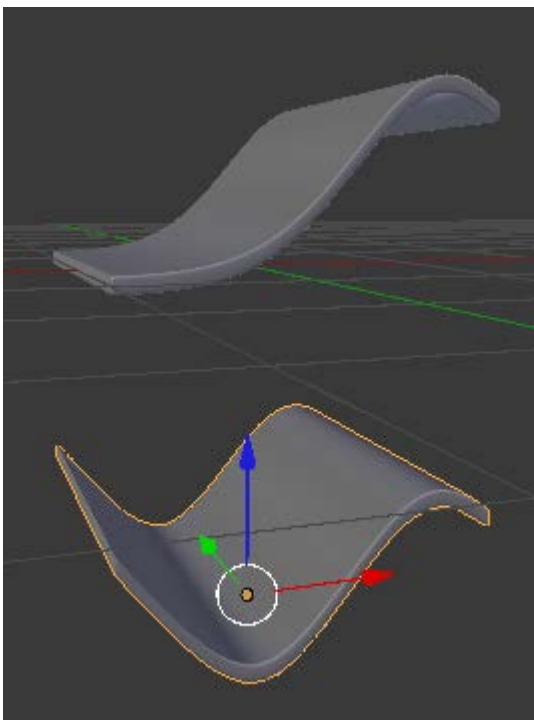


A este segundo duplicado le cambiamos el parámetro **Ancho** del modificador **Ola**. La longitud de onda del color violeta es más estrecha. Variamos:

- **Ancho**. De 2.00 a 1.00.
- **Estrechez**. De 0.85 a 1.64.

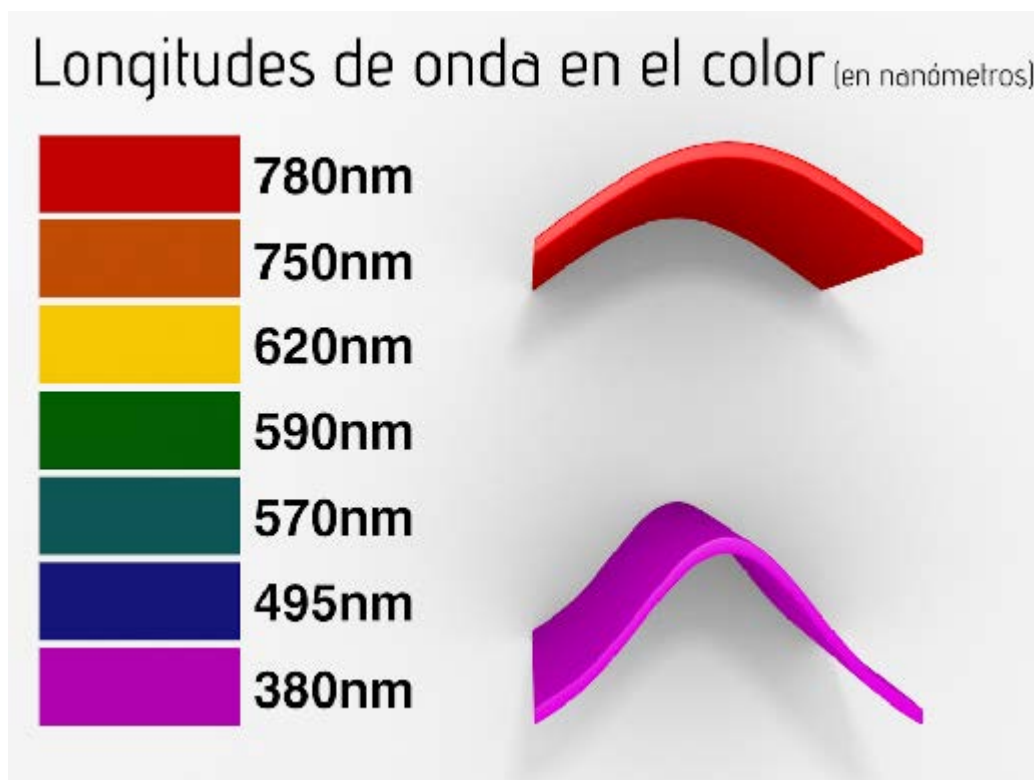



Puede que al poner en marcha la animación ("**Alt\_A**") parezca de que esta nueva longitud de onda va más rápido que la de arriba pero es sólo una ilusión óptica. La prueba es que no hemos alterado el parámetro **Velocidad**.

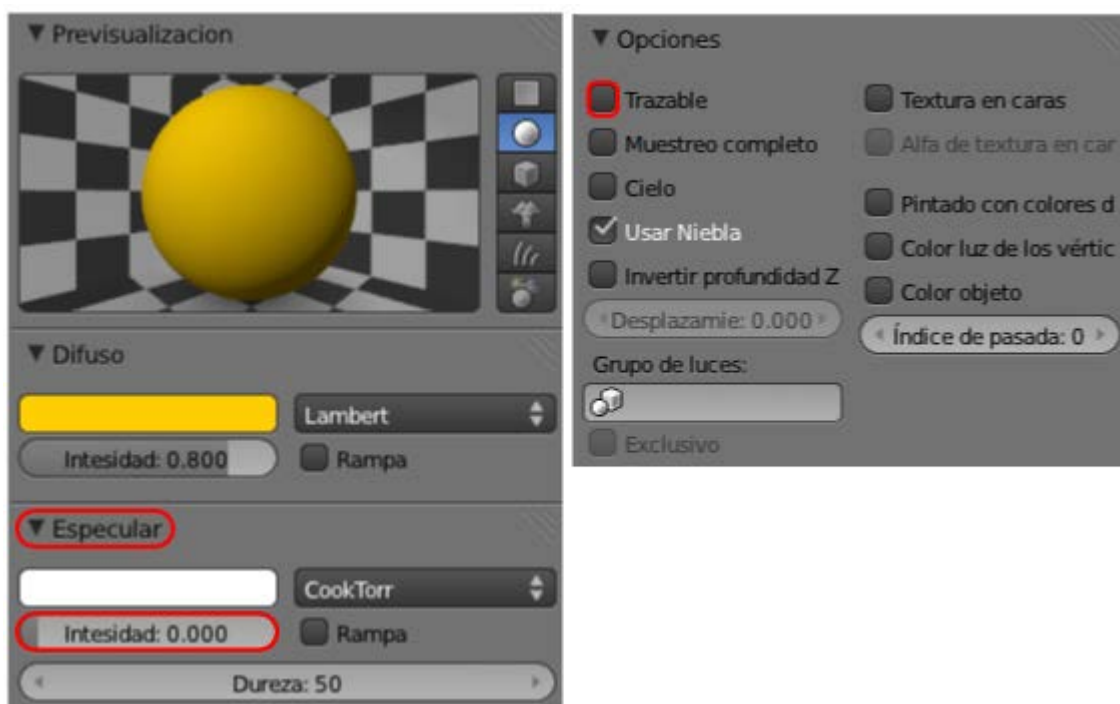


Al objeto de arriba le asignamos un color **FF0000** y al de abajo un **E700E5**.


El resto del trabajo se resume en crear la escena final.

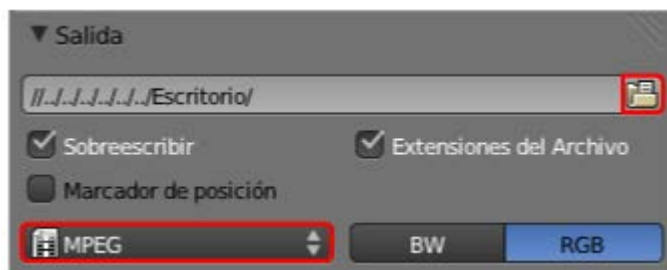


Todos los objetos del fondo tienen **desactivada** la opción **Trazable** en la botonera **Opciones** en su correspondiente **Material** . Esto hace que no proyecten sombra sobre el plano de fondo. También tienen el valor de **Intensidad** de **Especular** a **0.000**. Todo esto es para que el fondo sea lo más homogéneo posible y no se generen ni brillos ni sombras que impidan la lectura de la información. Un ejemplo:



Lo iluminamos con la habitual **iluminación básica** y pasamos a preparar la animación final en el panel **Render** .

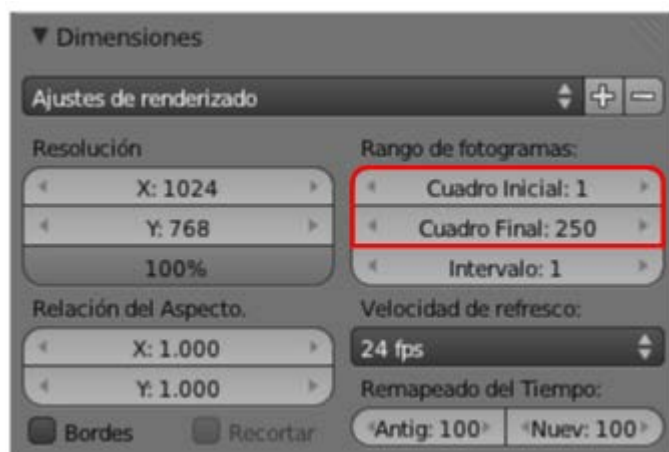
- La **ruta de destino** para el vídeo final. Usamos como es habitual el icono  de la botonera **Salida**.
- El **formato** de salida. Optamos por **MPEG** con el que se obtienen vídeos de buena calidad y muy poco peso.



- En la botonera **Configuración** nos aseguramos de que se generen archivos **MP4 (MPEG-4)** tal y cómo se indica en la siguiente imagen.



- **Número de fotogramas.** En la botonera **Dimensiones** decidimos el fotograma de inicio (**Cuadro inicial**) y el último (**Cuadro final**). Por defecto Blender trae esa configuración (**1-250**) pero podemos cambiar a nuestro gusto. Nosotros no lo cambiamos. Como la **Velocidad** de refresco, situada algo más abajo va a ser de **24fps** (*frames per second* - fotogramas por segundo) estaremos creando una animación de unos 10,5 segundos.



Tras elegir la **Resolución**, preparamos bien el encuadre y damos la orden de generar los 250 fotogramas con el botón **Animación**.



El premio a nuestro trabajo (y paciencia con los *renders*) lo encontramos en la carpeta de destino tras finalizar el proceso.



**Ayuda visual**



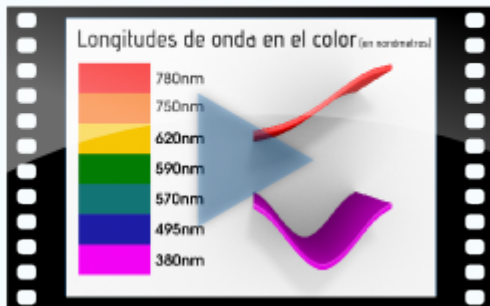
## Vídeo-tutorial Guardar la animación



## Resultado final



## Vídeo Longitudes de onda en el color



## 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.



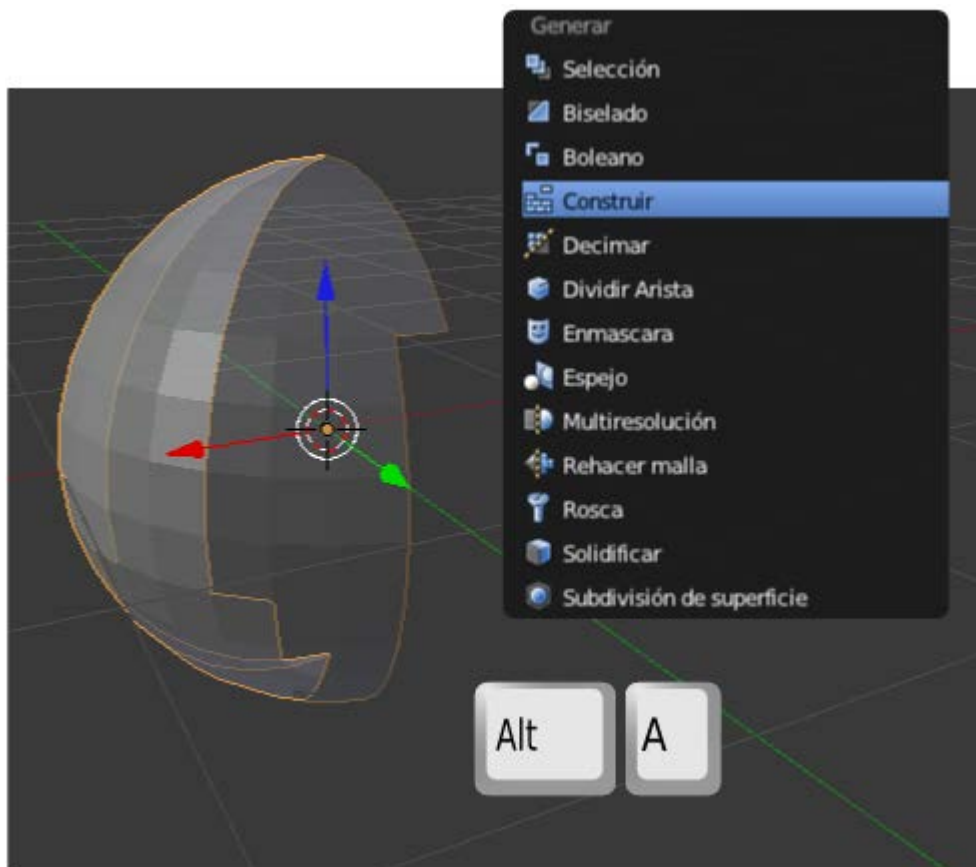
Archivo  
longitudes\_de\_onda.blend

## Construir

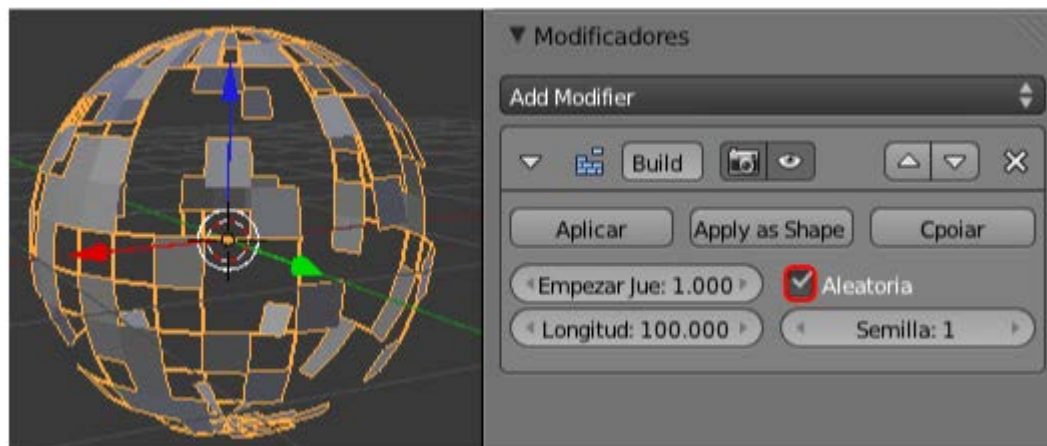
El modificador **Construir** puede dar resultados muy interesantes para embellecer presentaciones, cabeceras... y también genera una **animación directa**. Al igual que en el modificador **Ola** sólo nos preocupamos de jugar con sus opciones.

Si a una esfera UV (**Añadir/Malla/Esfera UV**) le aplicamos el modificador **Construir** desaparece por completo a la espera de que activemos la animación ("**Alt\_A**"). En ese momento comienza la construcción de la esfera que va apareciendo progresivamente.





Se trata de una aparición ordenada, que interesa sólo en determinadas ocasiones; la parte mas creativa se encuentra en la opción **Aleatoria** que consigue una construcción desordenada.



Otras opciones que incorpora:

- **Semilla**. Determina aleatoriamente el crecimiento de la construcción.
- **Empezar juego**. Si le damos un valor negativo y menor que el de **Longitud** la animación comienza con algo del objeto ya construido.
- **Longitud**. El número de fotogramas que le lleva al objeto formarse por completo. A valores altos, más lento es el efecto.

El recurso alcanza su verdadera fuerza estética si las caras de la malla son muy pequeñas



### Importante


Ya sabemos que la longitud de una animación por defecto en Blender es de 250 fotogramas por lo que en el modificador **Construir** tendremos esto en cuenta; o bien no sobrepasamos este valor o bien aumentamos aquellos 250 para que el efecto quede completo.






## Ampliando conceptos

Aprovecharemos las prácticas que hagamos con este efecto para probar algunos conceptos importantes en el tema de la animación:

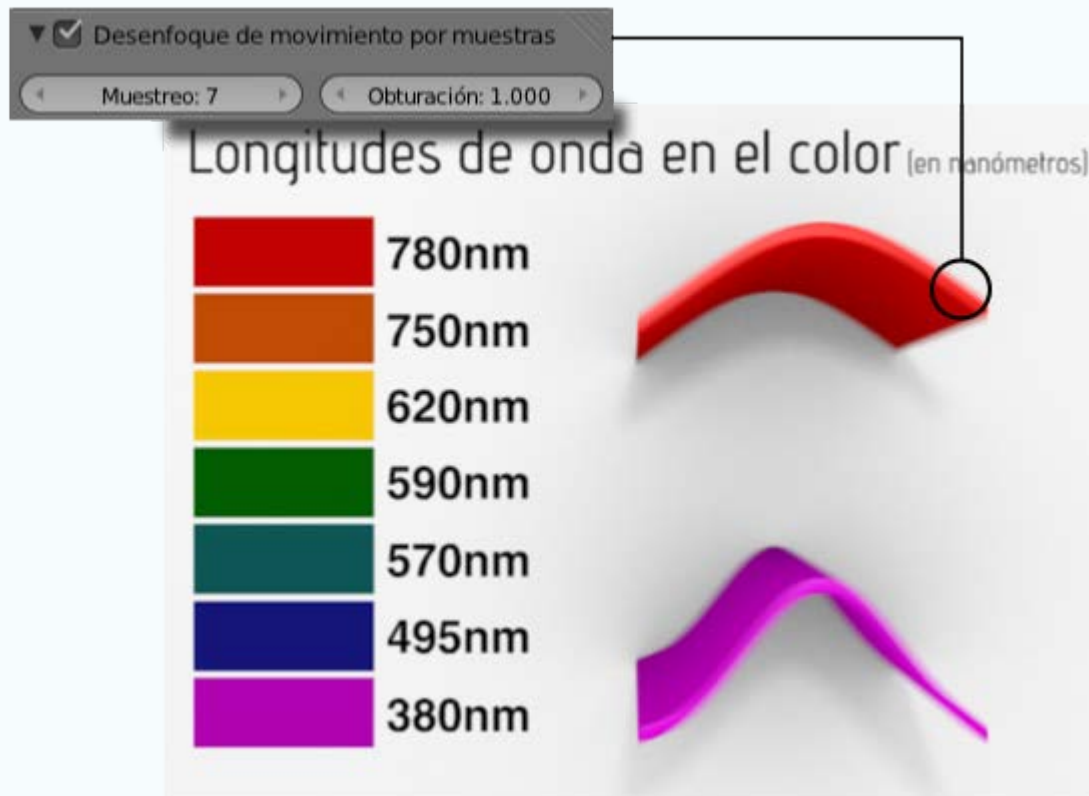
- Tanto para *renders* estáticos como para animaciones jugamos con la calidad del efecto **Anti-dentado** (*antialiasing*). Desde el comienzo de los tiempos de los gráficos por ordenador el efecto de **dientes de sierra** en los contornos de los objetos ha sido uno de los principales problemas. Para solucionar este asunto se usa la técnica *antialiasing* que difumina los contornos minimizando el problema. En la botonera **Anti-dentado** del panel **Render**  aumentamos o disminuimos la calidad de este recurso. Un aumento en este sentido supone un aumento en los tiempos de *render* pero la mejora en la calidad de la imagen es considerable.



- A través del editor **Línea de tiempo** , desplazando la línea vertical verde, nos movemos a cualquier punto de la animación para obtener un *render* en particular. Nos referimos a un *render* estático ("F12").



- Si queremos, por algún motivo, obtener todas las imágenes de la animación, procederemos como en 3DNP: crear una carpeta y escoger como formato de salida PNG, JPEG, o cualquier otro propio de imágenes estáticas.
- Tanto para imágenes fijas como para animaciones existe una técnica llamada **Desenfocado de movimiento** que se activa y configura en la botonera del mismo nombre en el panel **Render** . Consume muchísimos recursos pero en ocasiones es la diferencia entre un efecto bueno y otro espectacular. Cuando hablamos de *renders* de imágenes fijas nos referimos a fotogramas de una animación. Si aplicamos este recurso a una escena cuyos objetos no están en movimiento el efecto será nulo.



### Ayuda visual



### Vídeo-tutorial Construir

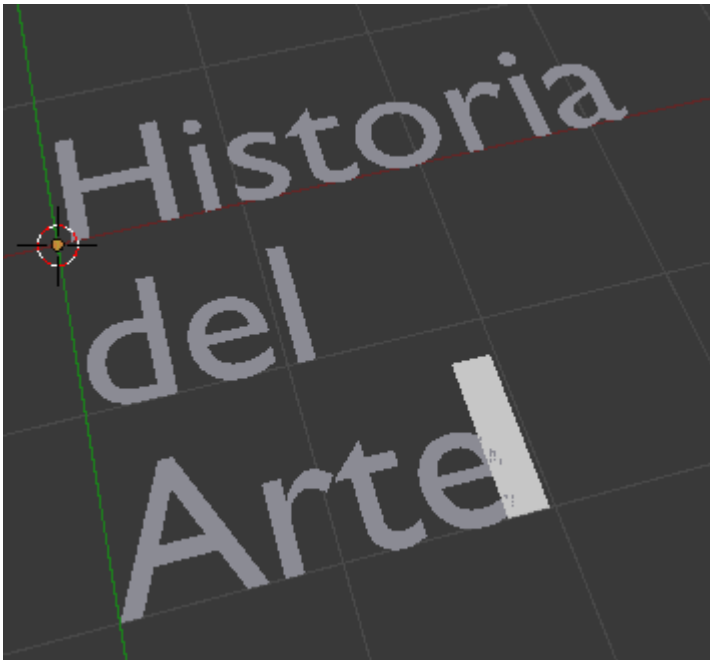


## Material didáctico: Cabecera de vídeo

Un vídeo siempre se merece una bonita cabecera para causar buenas sensaciones nada más comenzar.

Blender es un buen aliado para esos fines. En la práctica que viene a continuación hacemos una interesante cabecera con el texto "Historia del Arte".

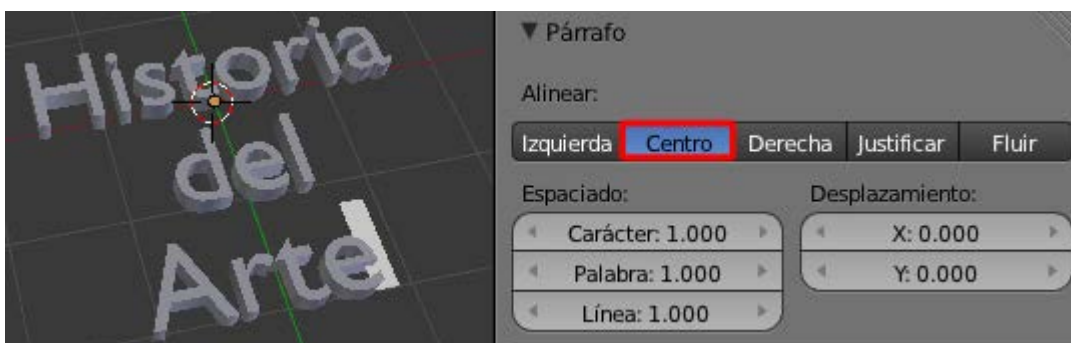
Eliminamos el cubo y ponemos un texto en su lugar (**Añadir/Texto**). Desde **Modo Edición**  escribimos la cabecera en cuestión.




En su panel **F** le aplicamos una **Extrusión** de valor **0.100** en la botonera **Geometría**.

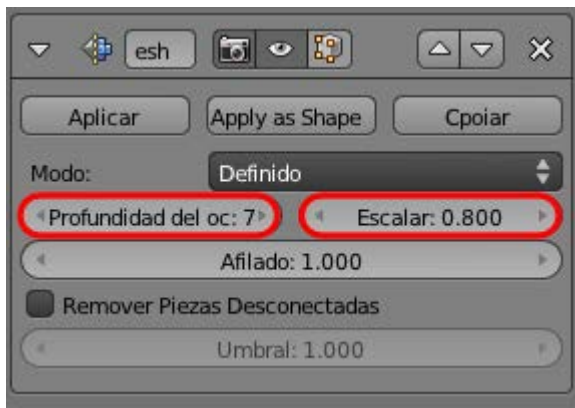


Alineamos en el centro usando la botonera **Párrafo**.



Es el momento de convertir, desde **Modo Objeto** , el texto en malla con **Texto/Convert to/Superficie desde curva-meta-texto**.



Sabemos perfectamente que esto deja una topología ruinosa, que arreglamos aplicando un modificador **Rehacer malla** con los parámetros que se indican en esta imagen.



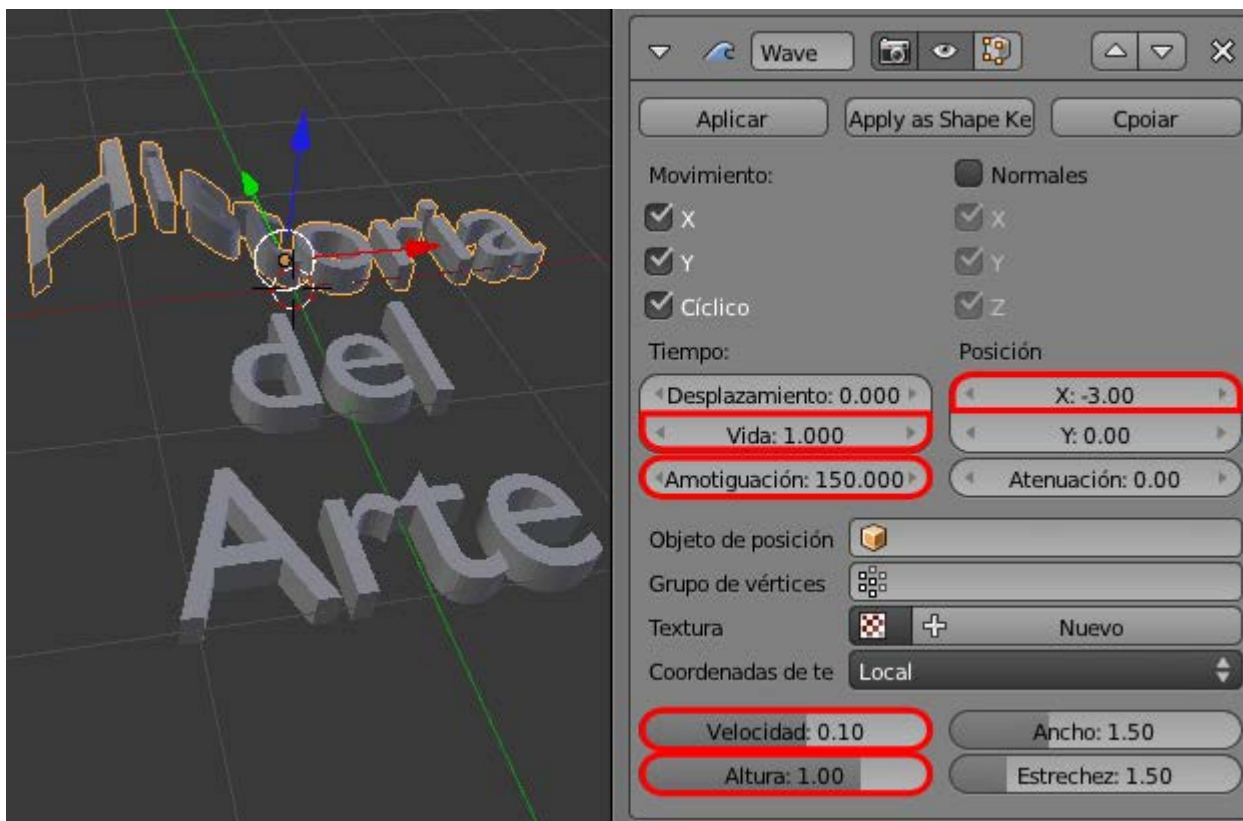
Tras **Aplicar** en **Modo Objeto** , esta es la apariencia en **Modo Edición**  (la topología es considerablemente mejor).



Una de las ventajas de **Rehacer malla** es que no hay que preocuparse de los **puntos dobles**.

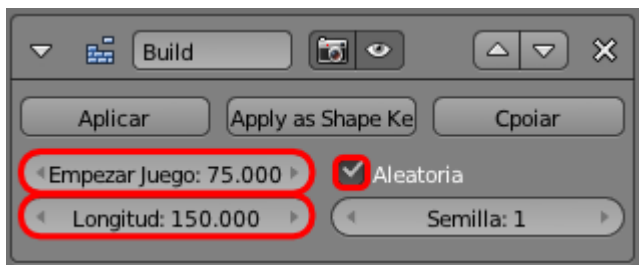
Hay que separar las tres palabras. Lo más cómodo es seleccionar todos los vértices ("A") en **Modo Edición**  y separar con "P"/**Por partes perdidas** y después, en **Modo Objeto** , seleccionar todas las letras de una palabra y hacer "Control\_J" (**Objeto/Unir**). Esto nos da como resultado tres objetos finales. Antes de seguir los seleccionamos y hacemos **Origen/Origen a geometría** en el cuadro **Herramientas** ("T").

Comenzamos por añadir a la palabra "Historia" un modificador **Ola** con estos parámetros.



- **Vida.** Comienza moverse en el fotograma 1.
- **Amortiguación.** Durante **150** fotogramas después de que termine **Vida** el efecto va descendiendo hasta extinguirse.
- **Posición X: -3.00.** Es una cuestión estética.

Ahora le añadimos un modificador **Construir** con estos parámetros:



- **Empezar juego: 75.** Para que no comience a verse el efecto hasta ese fotograma. Esto hace que la palabra sea invisible hasta ese momento.
- **Longitud. 150** para hacer coincidir su final con el del modificador **Ola**.

Con esto queda acabada al edición de la palabra "**Historia**", salvo por el **Material** que le queramos diseñar.

Para concluir debemos hacer lo mismo con las otras dos palabras con los siguientes cambios:

- Palabra "**del**". En el modificador **Ola** tendrá una **Vida** de **100** para que comience en ese fotograma. Y en el modificador **Construir** tendrá un **Comienzo de juego: 125**. Con todo esto conseguimos que el efecto comience justo antes de que termine el de la palabra "**Historia**".
- Palabra "**Arte**". En el modificador **Ola** tendrá una **Vida** de **175**. Y en el modificador **Construir** tendrá un **Comienzo de juego: 200**.


En realidad nada nos impide escalar y desplazar los elementos para conseguir una composición más armónica.



Por las condiciones de los modificadores esta animación debe tener al menos **400** fotogramas.



## Exportación (recordatorio)

Esta imagen incluye lo necesario para una buena exportación. Todo se edita en **Render** .





## Resultado final



### Video Cabecera



## 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  
cabecera.blend


## Seguir una trayectoria

Que un objeto siga una trayectoria es un clásico como ejercicio de iniciación a la animación. Su uso más frecuente es el de una cámara

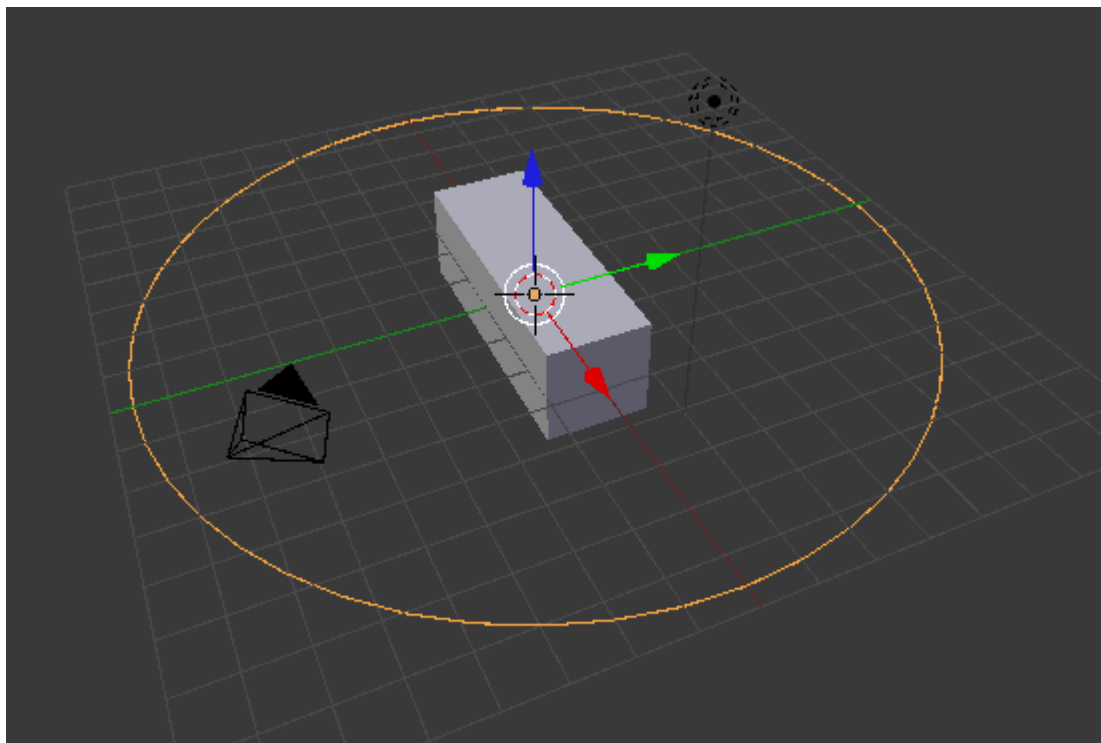


que se desplaza para conseguir un efecto cinematográfico de *travelling*, aunque hay otras innumerables ocasiones para echar mano de este tipo de animación.

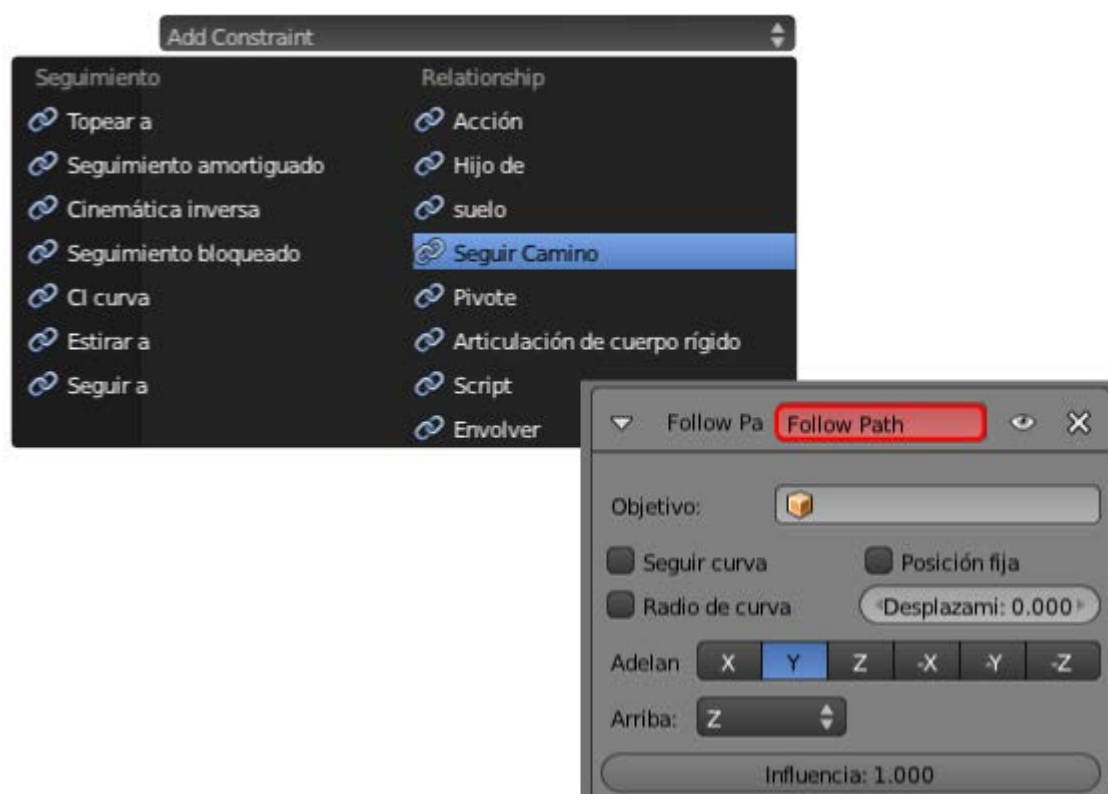
## La restricción

En el terreno de la animación es muy habitual que los objetos cumplan con obligaciones del tipo **no pasar de...**, **no girar más de...**, **apuntar hacia...**, **copiar el movimiento de...**. A ese tipo de obligaciones es lo que en Blender denominamos **Restricciones** y tienen su propio panel, cuyo icono es muy significativo: unas cadenas .


Para analizar el funcionamiento de este recurso necesitamos el cubo inicial y una curva circular (**Agregar/Curva/Círculo**). El cubo lo **escalamos en X ("SX")** y el círculo lo **escalamos ("S")** para que quede como en esta imagen.

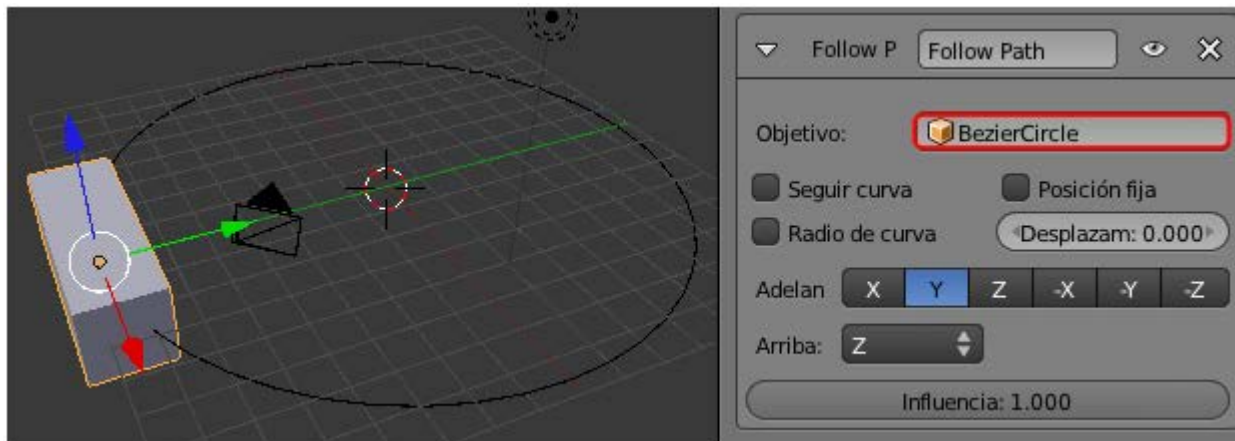


Al cubo le aplicamos la restricción **Seguir camino**.



Nada ocurre en el editor **Vista 3D**. El propio cuadro de opciones de la restricción parece estar alertándonos de que algo erróneo está pasando mediante un campo con el fondo rojo.

El motivo de que la restricción no tenga efecto es que Blender aún no sabe a qué curva debe seguir el cubo. En el campo **Objetivo**, con el icono , están las curvas disponibles (en nuestro caso sólo una llamada, por defecto, *BezierCircle*)




Ahora sí: el cubo obedece y se coloca en el punto que, a nivel interno, Blender considera el origen de la curva. Todo parece indicar que con la puesta en marcha de la animación ("**Alt\_A**") el cubo comenzará a desplazarse pero no es así. Pero sacamos nuestras propias conclusiones:

- **Adelante.** Nos dice que es el eje Y (**Local**) del objeto el que se coloca en dirección a la curva.
- **Arriba.** El eje Z (**Local**) es el que apunta hacia arriba.

Recordamos que al no haberse girado el cubo, sus ejes **Locales** coinciden con los **Globales** del entorno 3D.

## Una triquiñuela

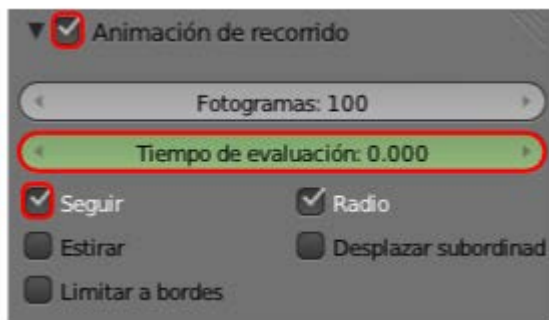
¿Por qué no hay movimiento si se pone en marcha la animación ("**Alt\_A**")?. Pues porque al contrario de los modificadores **Ola** y **Construir**, las restricciones no crean animaciones directas.

El problema radica en la curva. Si la seleccionamos y vamos a su panel  vemos que en la botonera **Animación de recorrido** (que tiene que estar activada) tiene un parámetro **Tiempo de evaluación**.



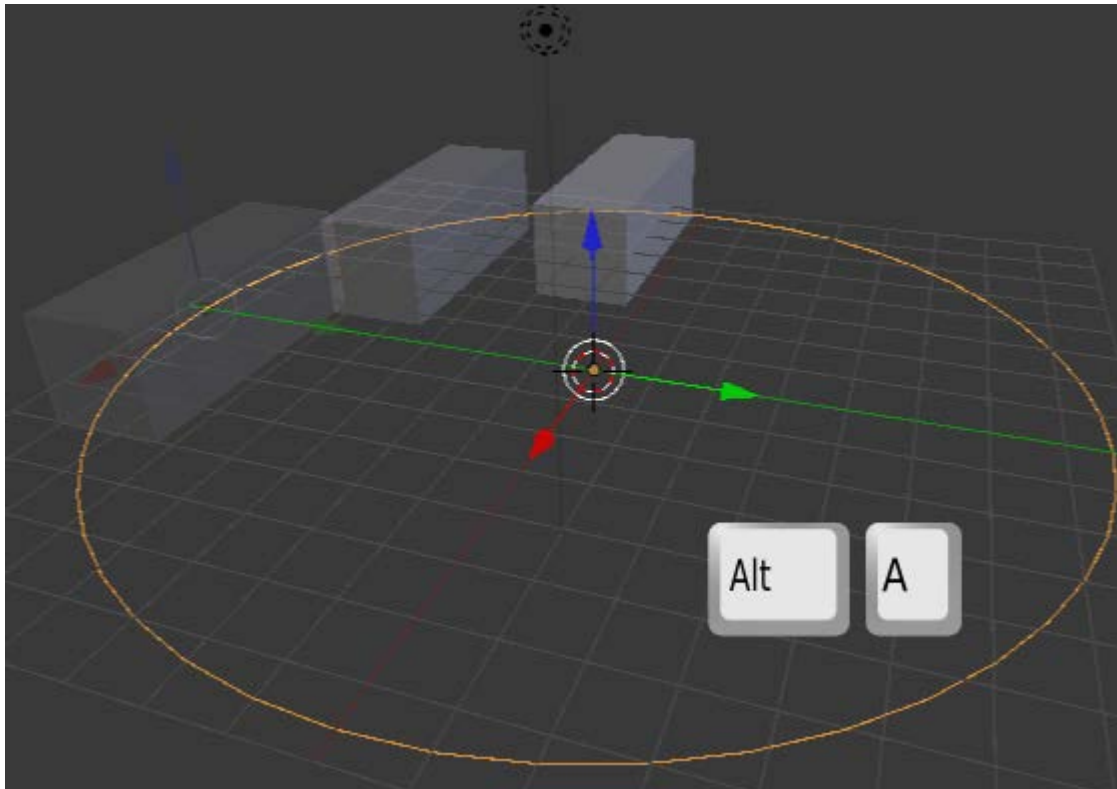
En teoría deberíamos crear nuestros primeros fotogramas clave ahí, pero nuestra intención en este módulo es no entrar en ese terreno; así que vamos a usar un truco para que se genere lo necesario y obtener la animación.

- Seleccionamos, en este orden, el cubo y después la curva para que esta sea el objeto dominante.
- Hacemos **Objeto/Padre/Establecer/Seguir camino** (o con "**Control\_P**"/**Seguir camino**). Ocurren tres cosas:
  - **El cubo se descontrola.**
  - Se ha **activado** automáticamente al opción **Seguir**. Sin embargo es intrascendente en nuestro plan; lo mismo da que esté activado o desactivado.
  - **Se crean los fotogramas clave** que necesitamos. La prueba es el cambio de color del parámetro **Tiempo de evaluación**.



- Eliminamos el parentesco con **Objeto/Padre/Limpiar/Eliminar padre** (o "**Alt\_P**"/**Eliminar padre**). Para esta operación al menos el cubo tiene que estar seleccionado. Curiosamente, y favorablemente a nuestros planes, la posición del cubo se restituye mientras que los fotogramas clave permanecen en el parámetro **Tiempo de evaluación**.

Con este sencillo truco tenemos nuestra animación. Ahora sí, al hacer "**Alt\_A**" el cubo comenzará a moverse.



### Ayuda visual



### Vídeo-tutorial "Seguir camino" (la triquiñuela)



## Configurar el movimiento

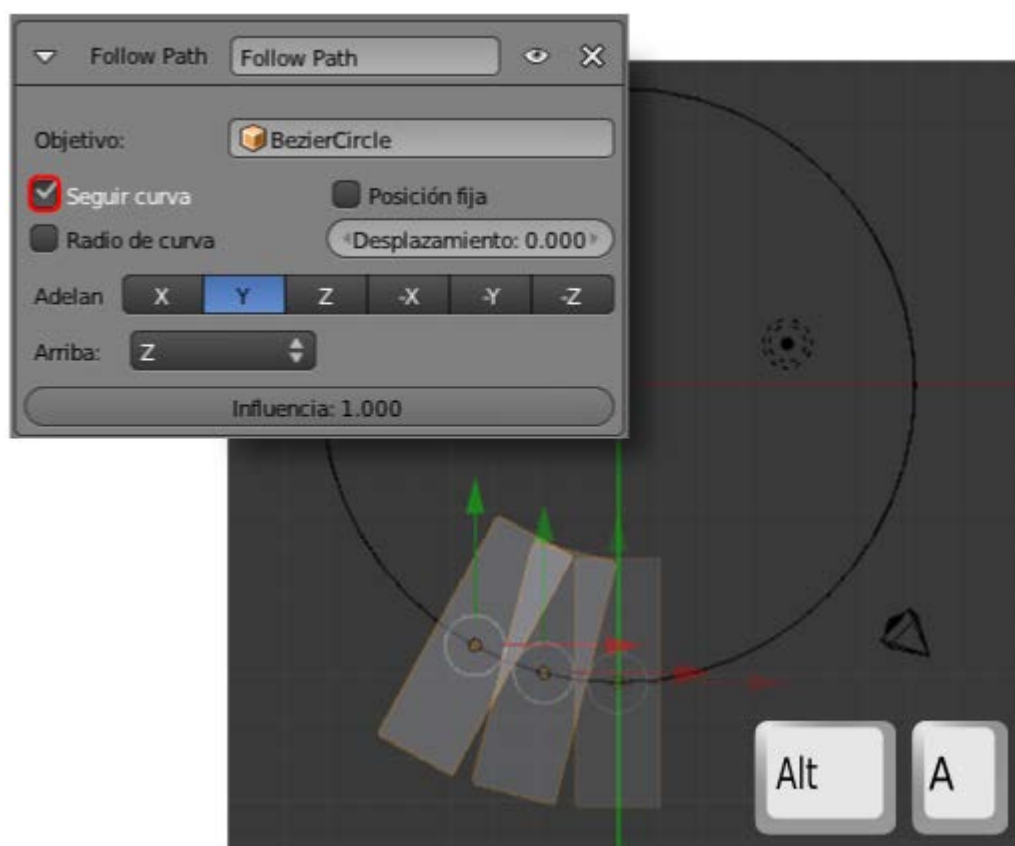
Al seleccionar la curva y acceder a su panel  volvemos a la botonera **Animación de recorrido** para determinar en número de fotogramas que necesita el círculo para ser transitado de principio a fin. Por defecto este valor es de **100**. Al igual que ocurría en el modificador **Construir** en el parámetro **Longitud**, a valores altos de **Fotogramas** menor será la velocidad. Nosotros alteramos el valor de Fotogramas a 250 para hacerlo coincidir con el valor por defecto de los fotogramas totales de la animación.



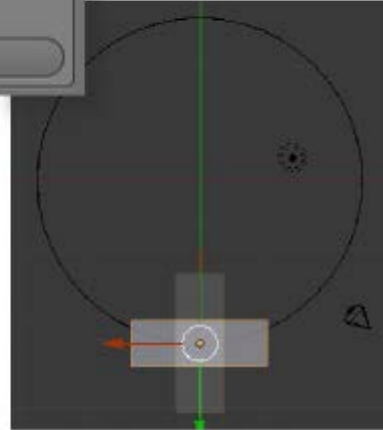
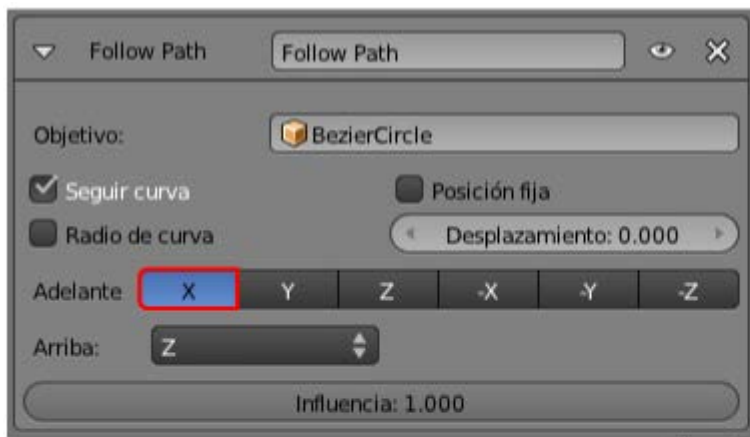
El cubo ahora sigue la trayectoria mucho más despacio que antes.

Para seguir configurando vamos a la **Restricción**  **Seguir camino** que tiene asignada el propio cubo:

- El cubo se está desplazando a lo largo de la trayectoria pero sin girar de acuerdo con ella. Al activar la opción **Seguir curva** la cosa cambia.



- Algo se ha ido al traste. Con esta orden los ejes **Locales** y los **Globales** ya no coinciden y ahora el eje **Y Local** mira hacia adentro de la curva. La solución es cambiar **Adelante** de Y a X.




### Ayuda visual

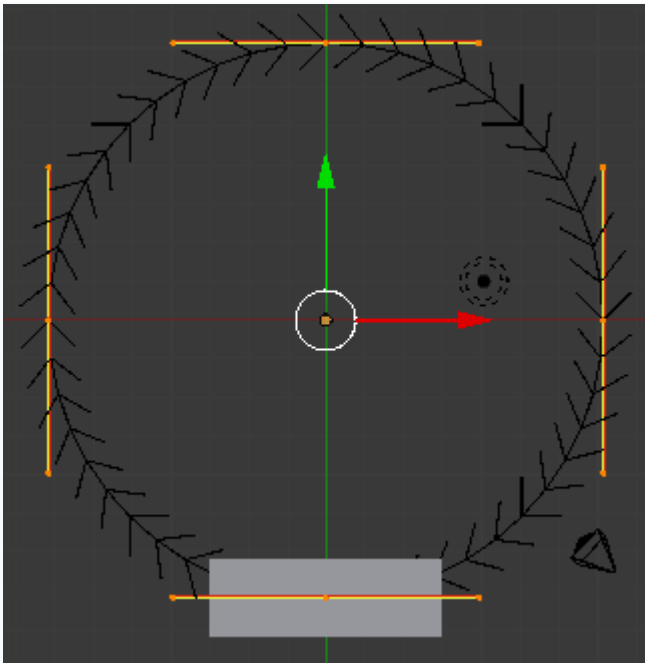


### Vídeo-tutorial "Seguir camino" (configuración)

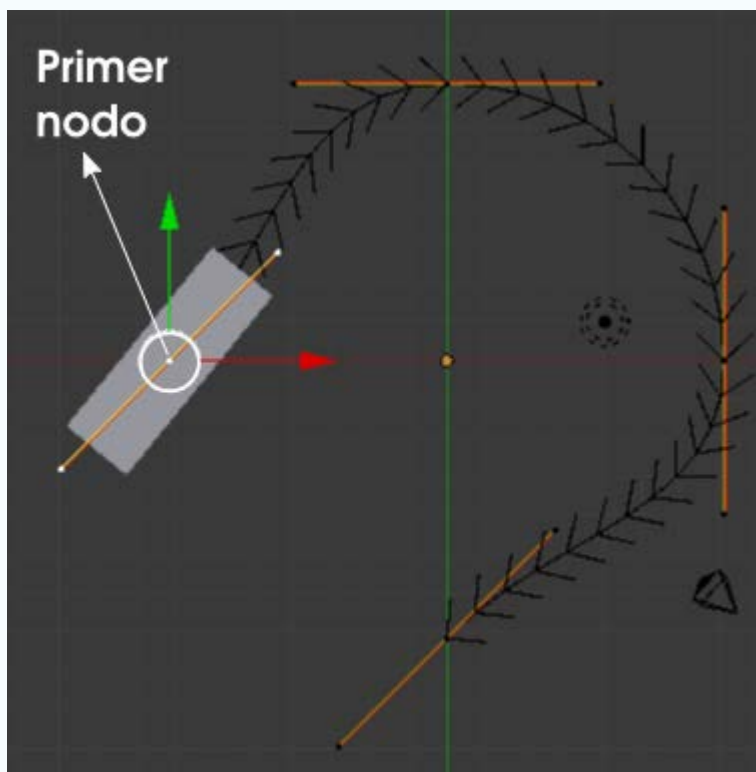


### Cambiar el sentido de la curva

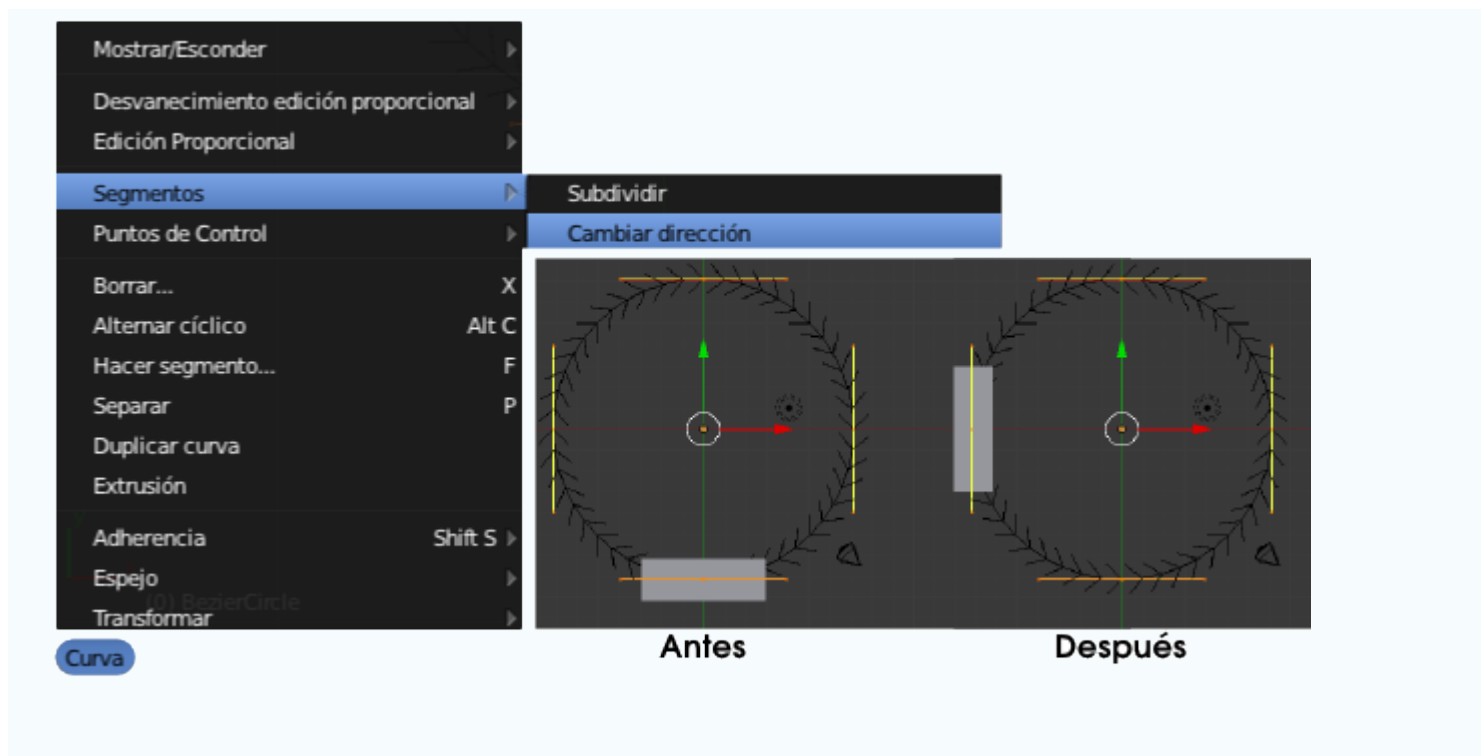
Mientras que en una curva abierta este cambio de sentido obedecerá a nuestras expectativas, en una curva cerrada ocurrirá un imprevisto. Este es nuestro ejemplo en **Modo Edición**  y los nodos seleccionados.



Si le decimos a Blender que represente esta curva abierta (**Curva/Alternar Cíclico**) descubriremos cuál es el primer vértice y cuál es el último.



Es una sorpresa, sin duda, y que nos explica por qué cuando a la curva cerrada le damos la orden **Curva/Segmentos/Cambiar dirección** este es el resultado.

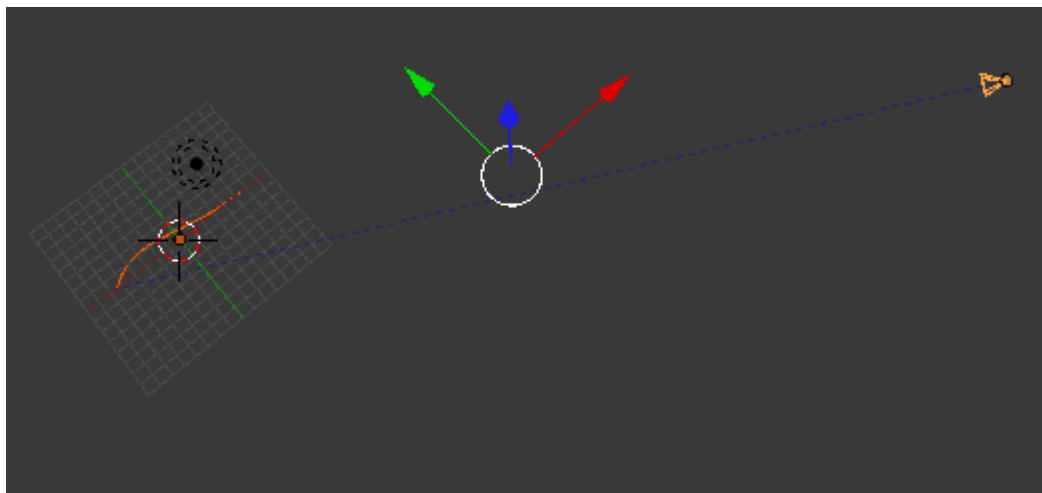


## El caso de la cámara

Suponemos que:

- Hemos añadido una trayectoria (**Añadir/Curva/Bezier**) y que la hemos **escalado ("S")** al alza.
- A la cámara le hemos aplicado la restricción **Seguir camino**.
- Hemos llevado a cabo la triquiñuela explicada para conseguir los *keyframes* de animación.

Esto es lo que tenemos.

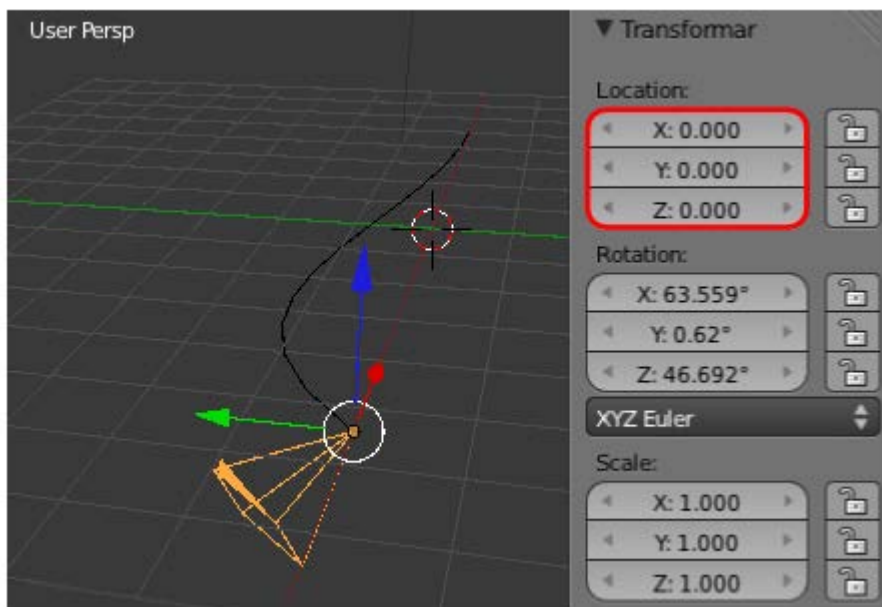


La cámara se comporta de un modo raro, alejándose de lo que debería ser su lugar de inicio en el recorrido. ¿Por qué la cámara se comporta de forma distinta a como lo hizo el cubo?

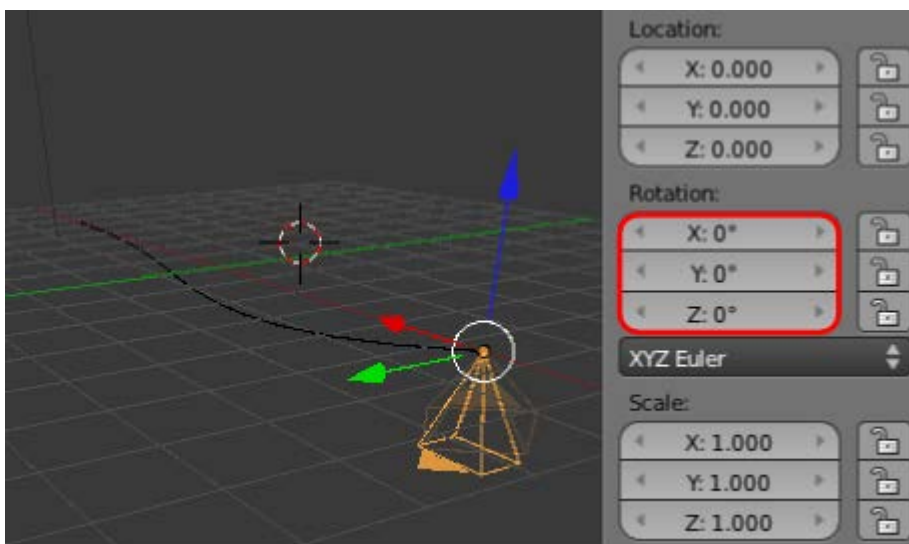
Vamos poco a poco. En la escena inicial de Blender la cámara no está en el origen de coordenadas e incluso aparece girada, de modo que los ejes **Locales** y los **Globales** no coinciden. Y es eso lo que está originando un resultado incontrolado. La solución.:

- **Eliminar el desplazamiento que ha sufrido el objeto.** Para ello vamos al cuadro **Propiedades ("N")** y en la botonera **Transformar** ponemos todos los valores de **Location** a **0.000** (otra forma es usar el menú **Objeto/Limpiar/Posición**). Con esto el origen de la cámara se sitúa al principio de la curva. No se coloca en el punto 0,0,0 del sistema **Global** porque tiene asignada la restricción **Seguir camino**.





- **Eliminar la rotación.** Podríamos usar, como antes el cuadro **Propiedades ("N")** y poner a **0.000** todos los parámetros **Rotation**, o usar el menú **Objeto/Limpiar/Rotación**.



## Los ejes de la cámara

Debe llamarnos la atención que una de las particularidades más significativas de la cámara es que su eje Z (azul) es el que queda perpendicular al encuadre; de esa manera los ejes X e Y guardan coherencia respecto al punto de vista de la cámara.

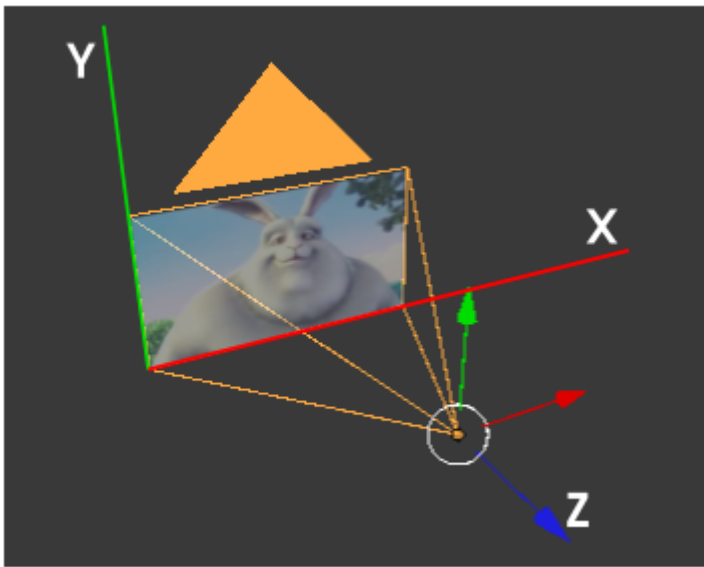
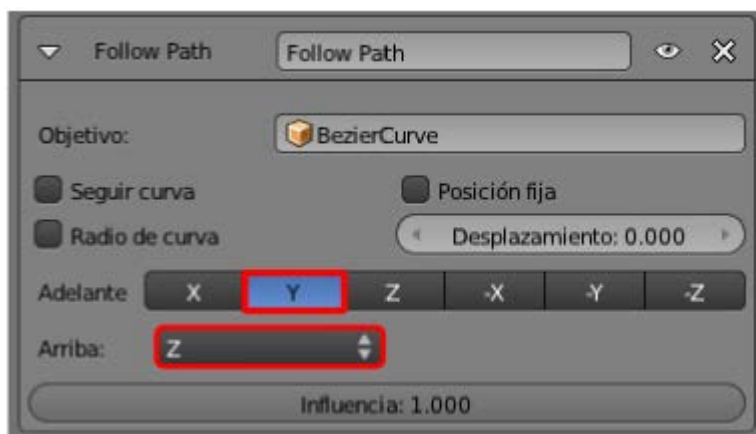


Imagen: Fotograma de Big Buck Bunny // Licencia: CC-BY-3.0

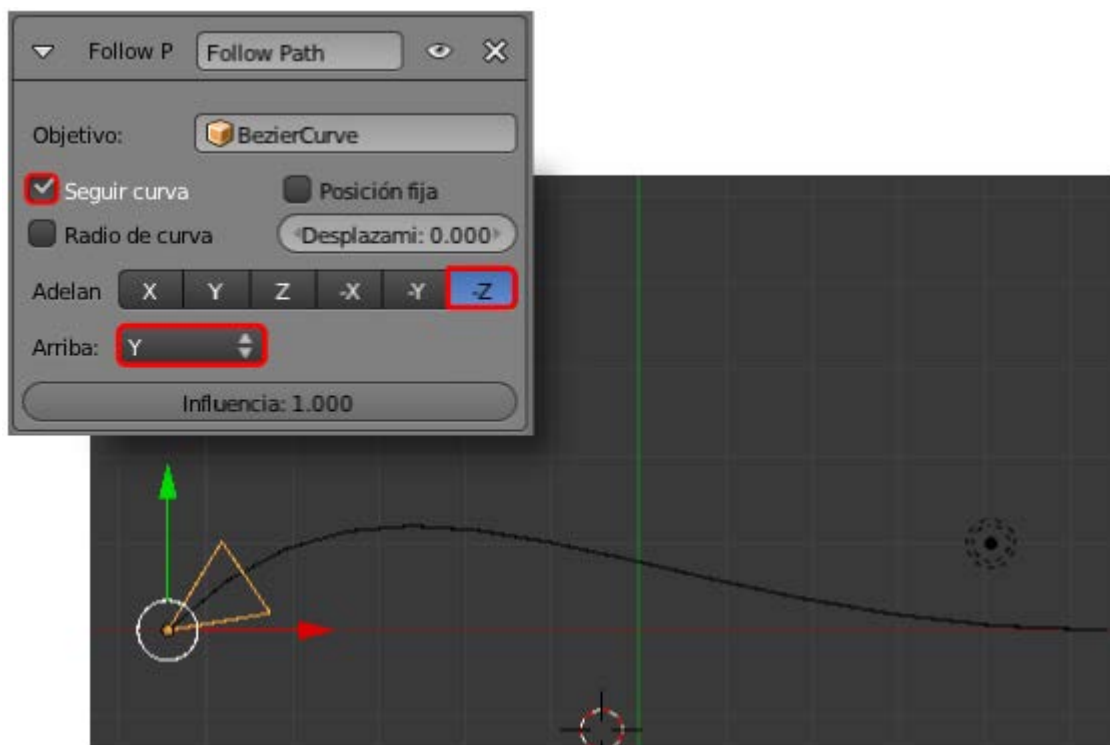
Y el tema de los ejes sigue condicionando la restricción **Seguir camino**. Por defecto esta restricción sabemos que orienta al objeto con:

- Adelante: Y.
- Arriba: Z.



Sin embargo esas opciones no son buenas para nosotros debido a la orientación de los ejes de la cámara. Debemos cambiar:

- Activar **Seguir curva**.
- Adelante: -Z (negativo).
- Arriba: Y.



### 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  
seguir\_camino\_camara.blend

## Seguir a


En el caso de la cámara puede que la opción **Seguir curva** se adapte a nuestras necesidades pero lo más habitual es una situación como la de 3DNP; es decir, que mientras la cámara se desplaza, ésta mira siempre a un punto concreto de la escena.

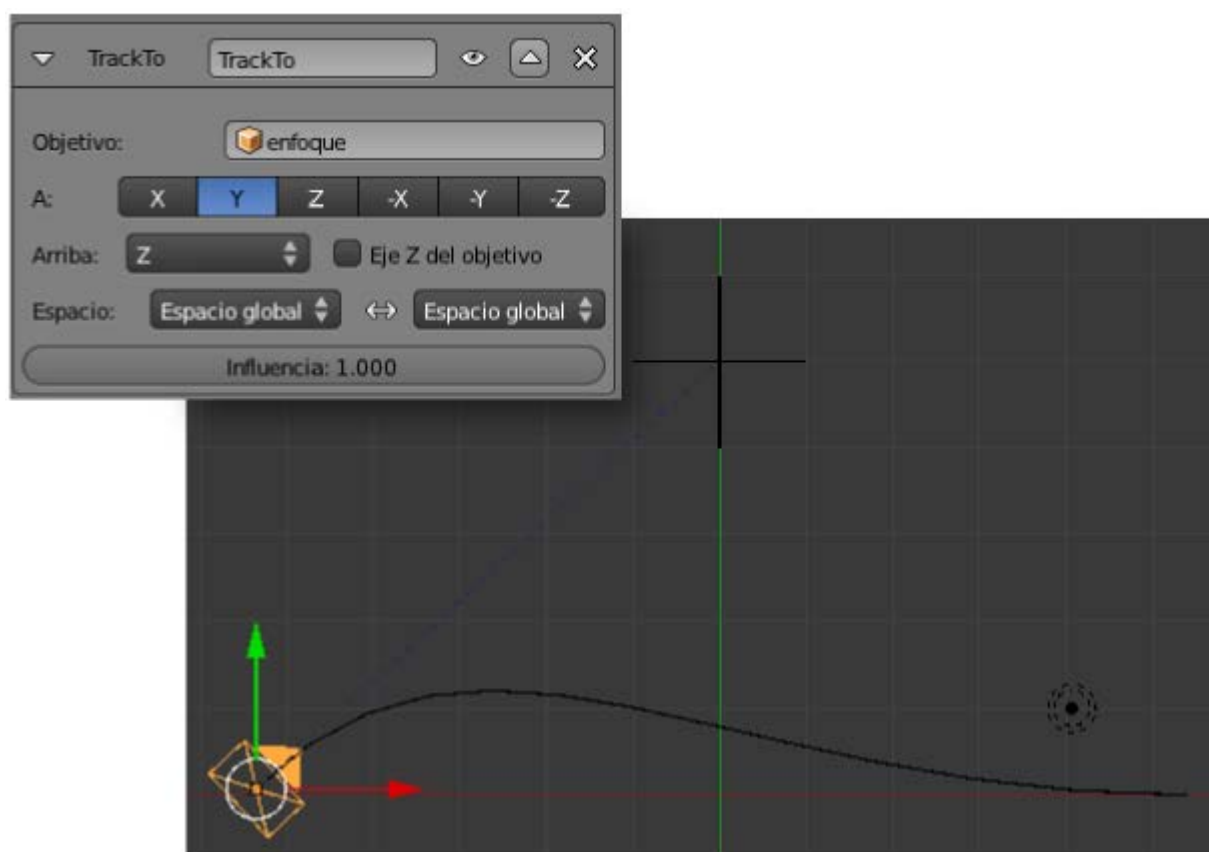
En ese caso echamos mano de una segunda restricción y, por norma general, de un objeto vacío (**Añadir/Vacío**) que hace las veces de punto de enfoque.



La idea aquí es que la cámara siempre mire a ese nuevo objeto que, en realidad, podría ser una lámpara o cualquier otro objeto de la escena; el hecho de añadir un **Vacío** es interesante para no involucrar a ninguno de esos objetos en esta edición.

¿Qué debemos cambiar en la configuración del apartado anterior?

A la cámara le asignamos una segunda restricción  de tipo **Seguir a** donde le escogemos como **Objetivo: Empty** (este es el nombre que Blender le ha dado al objeto **Vacío**, pero lo ideal es que nosotros le hubiéramos puesto un nombre apropiado como *enfoque*). Inmediatamente la cámara se disloca y aparece una línea azul punteada que representa la relación entre la cámara y **Vacío**.



A raíz de aplicar la restricción **Seguir a** resulta que la orientación de los ejes marcada en la otra restricción **Seguir camino** ha dejado de funcionar; es razonable que la cámara no pueda mirar a la dirección de la trayectoria mientras mira también al objeto **Vacío**.



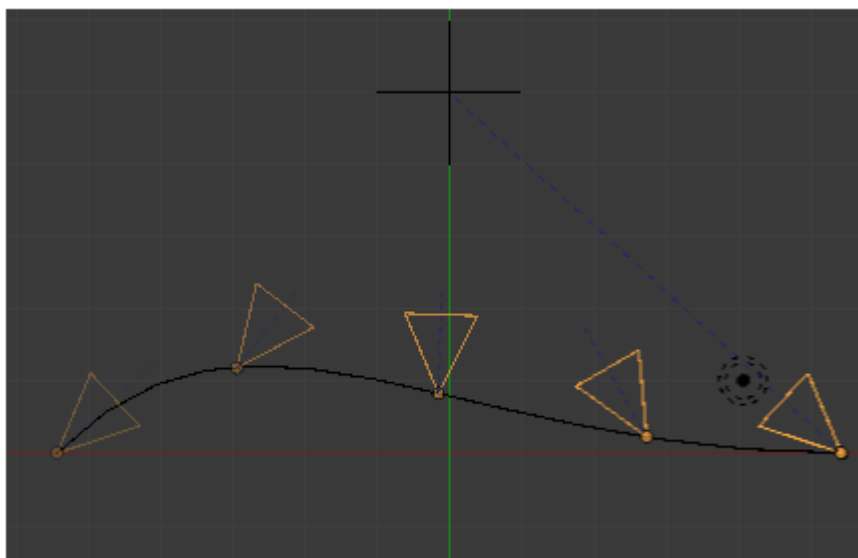
### Importante

Esto que ha ocurrido respecto a que los ejes elegidos en **Seguir camino** dejen de funcionar es consecuencia de que la restricción **Seguir a** (*track to*) está debajo y por lo tanto es la última que se tiene en consideración.

Así que no queda otra que volver a definir los ejes en la restricción **Seguir a** (*Track to*).



Al activar la animación, la cámara se verá obligada a mirar a **Vacío** sea cual sea la posición que ocupe en la trayectoria.



### Ayuda visual



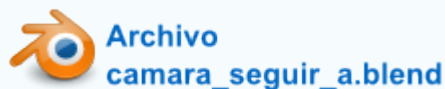
#### Vídeo-tutorial "Seguir a"





## 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.

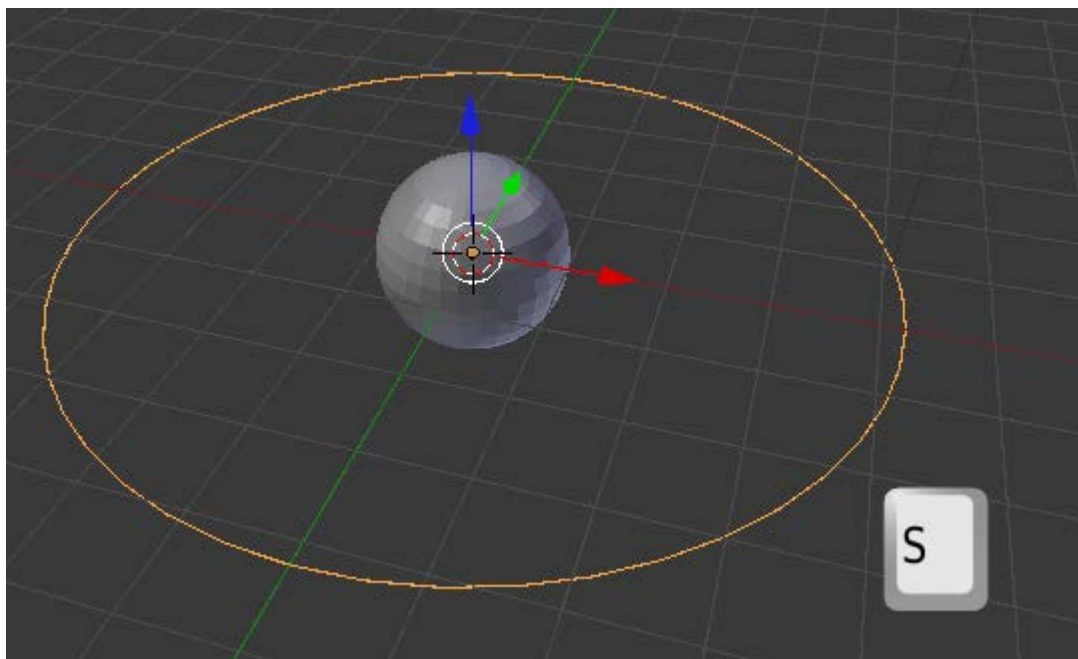


## Material didáctico: Fases de la luna


Pocas experiencias pueden compararse con explicarle a un niño por qué la Luna cambia de forma.


Vamos a crear una animación para ayudar en esa explicación aunque incluye una pequeña mentira: no es la Luna la que se mueve sino la lámpara.

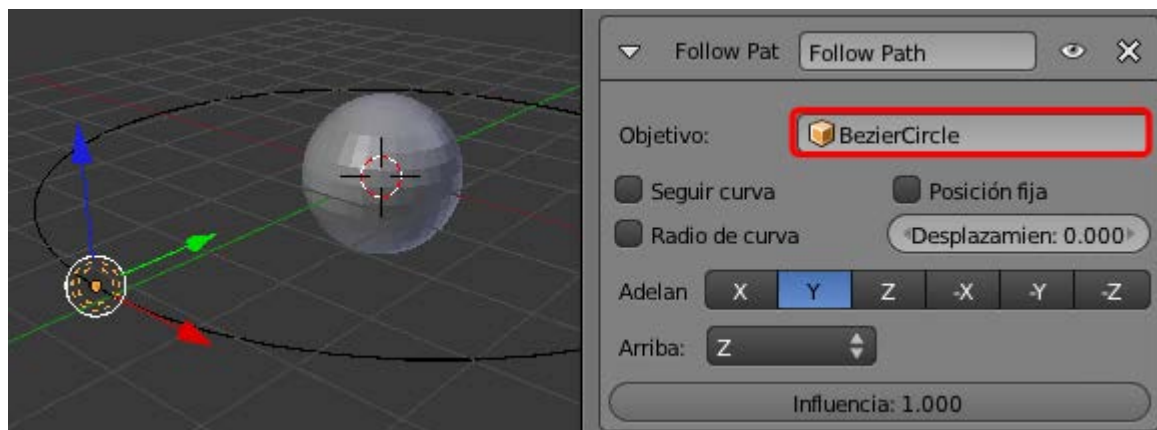
Sustituimos el cubo de inicio por una esfera (**Añadir/Malla/Esfera UV**) y aprovechando la correcta localización del **Cursor 3D** sacamos también una curva circular (**Añadir/Curva/Círculo**). Esta curva sale a escena con un radio igual que el de la esfera así que inmediatamente la **escalamos** ("S").



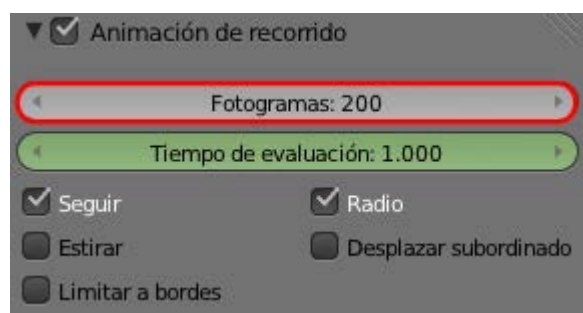
La escena ya cuenta con una lámpara y la vamos a utilizar para la animación. Seguimos el protocolo para que se originen los fotogramas clave con triquiñuela incluida:

- Seleccionamos primero la lámpara **Puntual**  y luego el círculo.
- Emparentamos con **"Control\_P"/Seguir camino**. Se crean los fotogramas clave.
- Desemparentamos con **"Alt\_P"/Eliminar padre**. Concluimos la triquiñuela.
- Eliminamos las transformaciones previas de la lámpara; en este caso **Objeto/Limpiar/Posición**. La lámpara se sitúa en 0.0.0 por lo que queda dentro de la esfera.

Nos resta añadir una **Restricción**  de tipo **Seguir camino** a la lámpara. No olvidaremos que es este el objeto que tiene que estar seleccionado y que la curva (*BezierCircle*, si no le damos otro nombre) es el **Objetivo**:



Seleccionamos la trayectoria y en el panel **Curva** cambiamos el valor de **Fotogramas** a **200** para que el desplazamiento de la lámpara sea más lento.



El resto son asuntos estéticos. Por ejemplo en el panel **Mundo** hacemos un par de ediciones:

- **Color de horizonte.** Un hexadecimal **000000**.

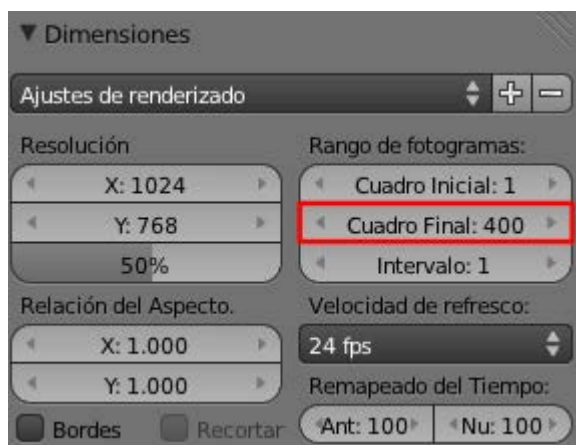


- **Estrellas.** Una configuración en esta línea (la esfera ya tiene un modificador **Subdivisión** y un sombreado **Suave** en el siguiente *render*; no está activada la **Oclusión ambiental**, la única iluminación es la de la lámpara a la que se le ajusta el valor de **Energía**).






Nos queda definir la longitud de la animación. **400** fotogramas es una buena opción porque de ese modo vemos dos veces el ciclo completo.



## Exportación (recordatorio)

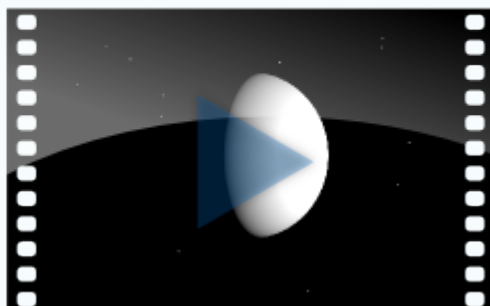
Esta imagen incluye lo necesario para una buena exportación. Todo se edita en **Render** .



## Resultado final



### Video Fases de la luna



## 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  
luna\_fases.blend

NOTA: este .blend incluye un mapeado de la luna.

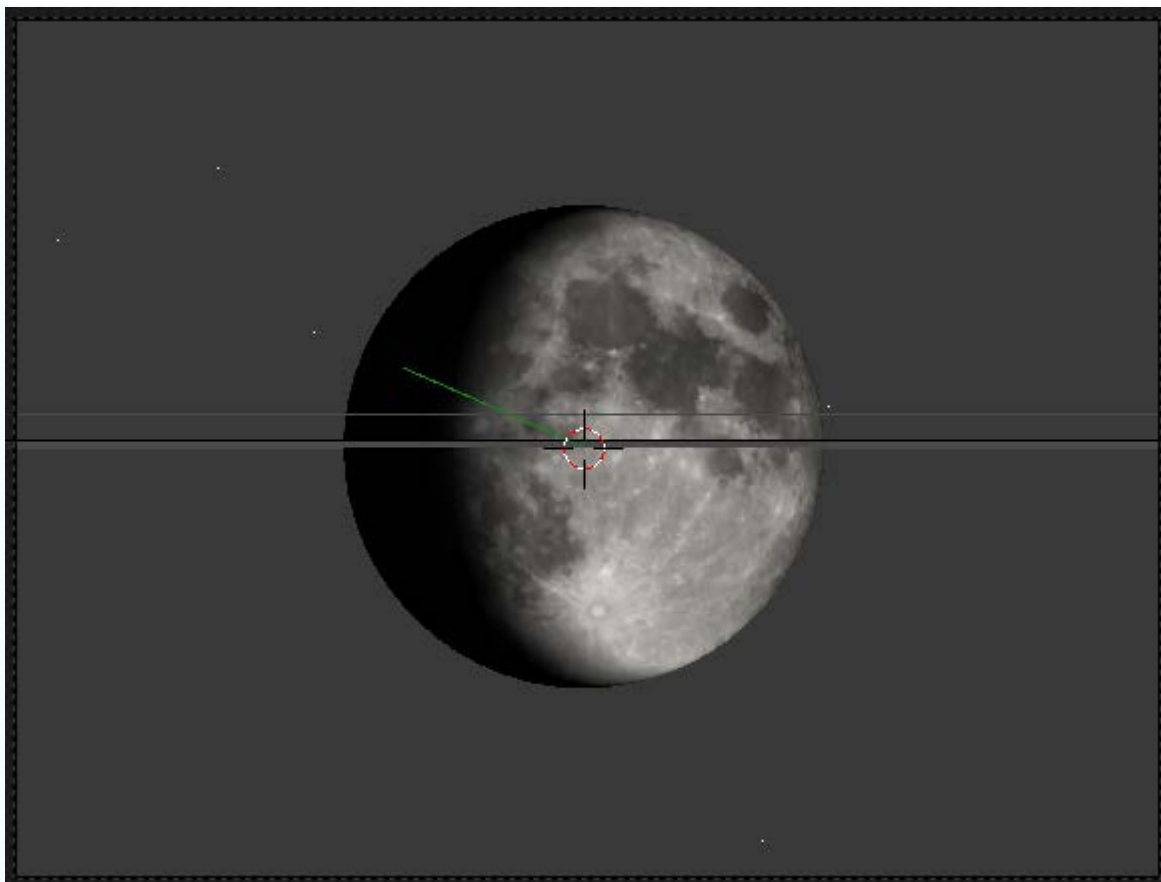


Imagen mapeada: Full moon // Autor: Luc Viatour // Licencia: GNU Free Documentation License

## Actividades



### 1- Varios navegadores

Comprueba la funcionalidad de 3DNP en distintos navegadores (Firefox, Chrome, Internet Explorer...)



### 2- Altera el javascript de 3DNP

No tengas miedo a alterar el *javascript* siempre que hagas una copia de seguridad. Lo recomendable es guardar una copia del ZIP entero.

Prueba alterando las variables que se han explicado y experimenta los resultados. Muchas veces los cambios supondrán que 3DNP no funcione bien, pero eso es indispensable para comprender su significado.



### 3- 3DNP en servidor remoto

Si tienes conocimientos sobre web sube tu producto derivado de 3DNP a un servidor remoto. Comprueba cómo la compresión de las imágenes favorece la carga del archivo y su navegación.



#### 4- Cámara en trayectorias

- Haz que la cámara siga curvas abiertas y cerradas. Altera el sentido de la curva para controlar hacia dónde va.
- Consigue que una cámara se desplace boca abajo.



#### 5- Otro objeto para "Seguir a"

- Haz que la cámara apunte siempre el encuadre a una lámpara.
- ¿Te atreves con este reto?. Haz que la cámara, mientras sigue una trayectoria, mire siempre a un cubo que se desplace por otra curva distinta.



#### 6- Velocidad de la cámara en movimiento

Coge destreza en manipular las características de la curva para controlar la velocidad de la cámara o cualquier otro objeto que se desplace por ella.



#### 7- La triquiñuela para "Seguir camino"

Repite, hasta que comprendas correctamente todo el proceso, la triquiñuela por la que se consiguen fotogramas clave para la curva sin crearlos manualmente.



#### 8- Otras restricciones

Investiga alguna restricción más aunque sea a ciegas. A ver si sale algo.

Un comienzo: **Suelo**.

- Saca un plano (**Añadir/Malla/Plano**) y un cubo (**Añadir/Malla/Cubo**)
- Al cubo aplícale una restricción **Suelo** con **Objetivo** el plano

Ahora trata de desplazar el cubo para colocarlo debajo del plano. No podrás.



## 9- Investiga a 3DNP

Accede al archivo *3DNP Blender.blend* y comprueba cómo la cámara, efectivamente, tiene asignada una restricción de tipo **Seguir a** (*Track to*).



## 10- 3DNP en vídeo

Cambia el formato de salida de **JPEG** (fijo) a **MPEG** (vídeo) en el archivo de **3DNP** y comprueba cómo obtienes en la carpeta **images** un vídeo no interactivo.

## Test de autoevaluación



### Autoevaluación: Interactividad y animaciones directas

**1- Si cambiamos las opciones que trae 3DNP por defecto, es casi seguro que debemos editar...**

El archivo *3DNP\_config.js*.

El archivo *3DNP\_loader.html*.

El archivo *3DNP.css*.

**2- El directorio de destino de las imágenes creadas en 3DNP debe ser una carpeta...**

De la que importa el nombre siempre que esté colocada junto a *3DNP.html*.

Llamada *3DNP* y colocada junto a *3DNP.html*.

Llamada *images* y colocada junto a *3DNP.html*.

**3- El producto obtenido con 3DNP está destinado a ser interactivo en...**

Un visor de imágenes.

Un navegador como Firefox o Internet Explorer.

El interior del propio Blender.

**4.- Para previsualizar una animación usamos...**

"Control\_A".

"Alt\_A".

"Shift\_A".

**5- ¿Para qué aumentamos el valor de Anti-dentado?**

Para conseguir *renders* sin efecto pixelado en los contornos de las formas.

Para conseguir sensación de movimiento en un *render* estático de una animación.

Para eliminar la textura facetada propia de algunos gráficos generados por ordenador.

**6- ¿Qué truco usamos para conseguir los fotogramas clave cuando un objeto sigue una trayectoria, y no tener que insertarlos manualmente?**

Unir los objetos y después separarlos.

Vincular los materiales y después desvincularlos.

Emparentar los objetos y después desemparentarlos.

**7- Para que la cámara apunte siempre a un punto en el encuadre, se recomienda usar...**

Un objeto Vacío

Una lámpara

Otra cámara

**8- En el panel propio de la curva, cuando es trayectoria de una animación, aumentamos el valor de Fotogramas para...**

Aumentar la velocidad del objeto que se desplaza.

Disminuir la velocidad del objeto que se desplaza.

Aumentar el tiempo de duración de la animación.

**9- ¿Puede una malla tener a la vez un modificador Ola y otro Construir?**

Nunca.

Sí.

Sólo si está primero Ola.

**10- Para traer objetos a 3DNP usamos...**

Archivo/Añadir.

Archivo/Importar

Archivo/Vincular