Centro de Profesorado
Luisa Revuelta
(Córdoba)

Introducción a los Sistemas Gráficos 3D con Guadalinex

Carlos González Morcillo (Carlos.Gonzalez@uclm.es)
http://www.inf-cr.uclm.es/www/cglez

Grupo de Investigación ORETO

Escuela Superior de Informática
Universidad de Castilla-La Mancha

Sesión 01. Introducción a la Síntesis de Imagen 3D

En los últimos años se ha experimentado un crecimiento muy importante en el mundo del diseño por computador, y en concreto de la síntesis de imágenes tridimensionales. Las necesidades de personal cualificado en este sector son notables y continúan creciendo unidas al cada día mayor ámbito de aplicación del diseño 3D. Estas necesidades, igualmente patentes desde hace años en el mundo del *Software Libre*, han sido cubiertas con aplicaciones de calidad y uso profesional. En esta sesión se estudiarán los conceptos clave para entender en qué consiste el movimiento del Software Libre, junto con el ciclo de trabajo en producciones 3D empleando herramientas libres como Blender, Yafray, Gimp...

I. Fundamentos del Software Libre

Entendemos como software libre aquel que se basa en el cumplimiento de (al menos) cuatro libertadres básicas:

- Libertad para **utilizar** el programa para lo que quieras.
- Libertad para **estudiar** el el programa (para poder realizar un estudio del programa es necesario disponer del *código fuente*; que es una descripción formal del programa en un lenguaje entendible por el programador).
- Libertad para **redistribuir** el programa, y compartir sus beneficios con quien tú quieras.
- Libertad para **mejorar** el programa y distribuir sus mejoras.

Logotipo de GNU
(GNU is Not Unix)

Si leemos con atención las libertades básicas anteriores, veremos que en ningún caso se menciona nada sobre el precio del software. Para que un programa sea considerado Software Libre, debe cumplir las condiciones anteriores. Existen multitud de licencias que son consideradas válidas para Software Libre. Una de ellas (y quizás la más extendida) es la GPL (Licencia Pública General) de la FSF (Free Software Foundation).

Es importante no confundir el software gratuito (habitualmente denominado *Freeware*), con el software libre, ya que las licencias *Freeware* no permiten estudiar y mejorar el programa (por no disponer del código fuente). Esto es un impedimento crítico, ya que no permite la generación de software basado en estos programas, e impide el desarrollo económico local.


TUX, la mascota de Linux

Tampoco es correcta (aunque sí está muy extendida) la distinción entre *Software Comercial* y *Software Libre*. Un programa Libre puede ser comercial; se puede obtener un beneficio económico de él, y es totalmente lícito (y recomendable) ganar dinero realizando adaptaciones y dando soporte de este software. Es más correcto denominar *Software Privativo* al software que no es libre, porque nos priva de nuestras libertades básicas.

Linux es el núcleo de un Sistema Operativo que se distribuye bajo una licencia de Software Libre. En realidad, cuando instalamos un Linux, estamos instalando el núcleo del Sistema Operativo y un conjunto de utilidades que forman parte del proyecto GNU (GNU is Not Unix). Es más correcto hablar de distribuciones de GNU/Linux, ya que están formadas por el núcleo del Sistema Operativo y las herramientas libres de GNU.

Existen muchas distribuciones de Linux; la mayoría basadas en Red Hat o Debian. Entre ellas podemos destacar Fedora (la antigua Red Hat de escritorio). Suse, Mandriva (Mandrake), Debian, Ubuntu (basada en Debian), MoLinux (basa en Ubuntu), etc... Todas tienen un núcleo de Linux común, y se diferencian únicamente en los asistentes para la instalación del sistema y la configuración de periféricos. Una aplicación de GNU/Linux debería funcionar en cualquiera de estas distribuciones.

Se puede encontrar multitud de Software Libre para otros sistemas operativos que no son Linux. De hecho, la mayoría de las aplicaciones libres de Linux se pueden encontrar también en Windows y Mac entre otros.

¿Por qué utilizar Software Libre? Existen multitud de razones; entre otras podemos citar:

- **El software libre favorece el Desarrollo Económico Local:** Las inversiones, en vez de realizarse "alquilando" licencias a grandes multinacionales, se realizan en empresas de desarrollo nacionales. Esto es claramente positivo para la comunidad.
- **El software libre favorece la Innovación:** No se repite mil veces lo mismo; los programadores sólo se ocupan de programar nueva funcionalidad.
- **El software libre es una clave Competitiva:** Utilizando software libre los usuarios siempre pueden estar actualizados, normalmente con coste cero. No hay problemas de licencias utilizando Software Libre.
- **El software libre nos hace ser más libres:** No existen formatos cerrados o propietarios. Las aplicaciones pueden compartir sus datos sin problema. Cada usuario puede utilizar el software que desee y, con seguridad, existirán conversores libres entre formatos de ficheros.

II. El ciclo de trabajo en Proyectos 3D

Podemos definir el ciclo de trabajo en producciones 3D estableciendo tres grandes fases de producción, cada una de las cuales tendrá asociadas un conjunto de tareas (algunas de estas tareas pueden no estar presentes en ciertos proyectos). Las fases de producción suelen completarse secuencialmente, y normalmente no hay que ejecutar tareas de una fase anterior, cuando ésta se ha completado. Las tareas dentro de una fase de trabajo siguen un orden aunque ajustes en tareas posteriores pueden requerir cambios en tareas que estaban completas. Estas tareas suelen desarrollarse en paralelo en grandes proyectos.

La etapa de **preproducción**, conlleva todas las tareas de planificación y especificación del proyecto. Incluye algunas tareas no visuales como la escritura del guión y la gestión del personal que interviene en el proyecto y otras tareas visuales como la creación del Storyboard y el desarrollo visual del proyecto.

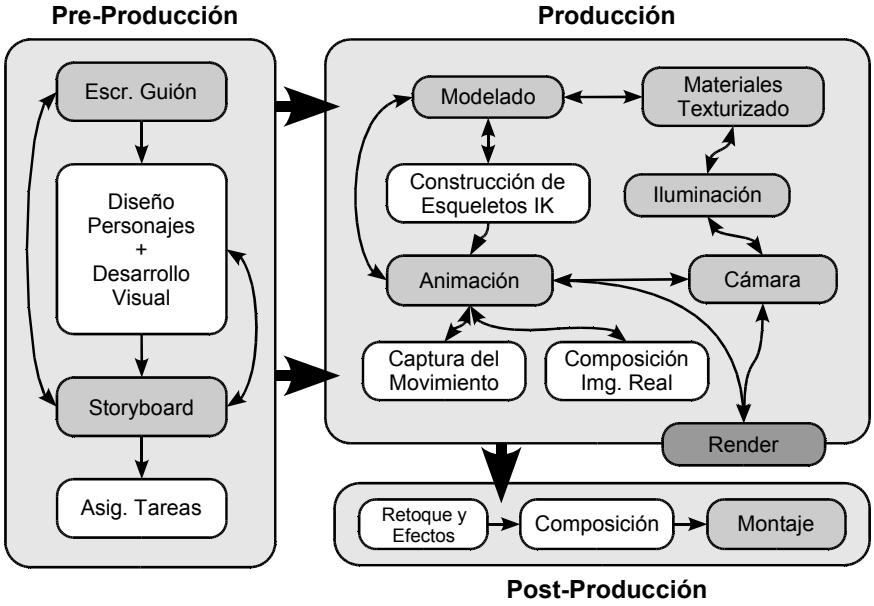


Imagen 1: *Fases del proceso de Síntesis de Imagen 3D*

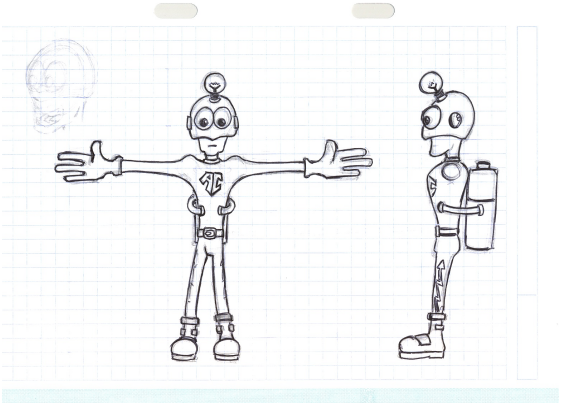
En la etapa de **producción** se llevan a cabo las tareas más costosas del proyecto. En primer lugar se realiza el modelado del personaje, construyendo su geometría de contorno. Una vez que los personajes y objetos están modelados, les asignaremos materiales y texturas, que darán una apariencia más realista al modelo. Podemos pensar en esta fase de texturizado como la encargada de asignarle una *"piel"* al objeto. Al igual que en el mundo real, para percibir algo del entorno necesitamos que nuestra escena esté iluminada. Tendremos que definir fuentes de luz virtuales que arrojen luz a nuestros objetos. Si el proyecto lo requiere, definiremos un esqueleto interno que utilizaremos para animar nuestros personajes digitales. Estos huesos servirán para definir el movimiento en las articulaciones de nuestros personajes sin preocuparnos de la geometría que las forma. La animación puede realizarse empleando sistemas de captura del movimiento que ahorran mucho tiempo a la hora de conseguir movimientos realistas y para ajustar movimientos compuestos con imagen real. Finalmente, acabaremos esta fase definiendo una cámara virtual que limitará la ventana de visualización de nuestra escena. Esta cámara, terminará de definir los parámetros necesarios para iniciar la etapa de generación de la imagen 2D de nuestro mundo 3D en el proceso conocido con el nombre de *render*.

La etapa de **postproducción** toma como entrada las imágenes generadas en la etapa de *render* de la fase anterior y las compone, aplicándoles una serie de filtros y modificadores antes de generar las imágenes definitivas en el formato de publicación final.

a) Preproducción

La fase de preproducción comienza con la **escritura del guión**. Tanto en pequeños como en grandes proyectos, suele ser la primera tarea, que será transformado visualmente en el storyboard. En pequeños proyectos suele estar clara desde el principio la magnitud de todas las etapas, mientras que en grandes proyectos es habitual encontrarse con limitaciones técnicas que requieran un cambio en el guión o en el storyboard.

Con el guión definido, el equipo de **desarrollo visual**, habitualmente formado por ilustradores, establecen la dirección visual y el estilo del proyecto. Se eligen los colores clave que complementarán visualmente las metas de cada parte de la historia. De igual forma, se desarrollan las **hojas de personajes**, con los bocetos del aspecto que tendrán los personajes que serán incluídos en el proyecto.



La fase de preproducción finaliza con la construcción del **Storyboard**. Mediante el *Storyboard* trasladamos el guión a imágenes. Estas imágenes serán tratadas como unidades que la fase de producción pueda gestionar como bloques

de trabajo. El *Storyboard* deberá centrarse en cómo contar la historia, composición de cámara, acciones... sin prestar atención a detalles técnicos. Aunque hay algunos convenios más o menos establecidos, cada autor define su propio formato de *Storyboard*. A menudo incluye notación para indicar los movimientos de cámara y dar así más riqueza a la narración visual.

b) Modelado

En esta etapa se obtiene una representación tridimensional de los objetos que intervendrán en la escena. Los paquetes software actuales permiten utilizar multitud de herramientas para el modelado de los objetos; incluso hay software especializado en esta etapa. La elección del mecanismo de modelado más adecuado puede ahorrarnos muchas horas. La mayoría de los programas permiten utilizar las siguientes herramientas y modos de trabajo:

- **Modelado poligonal básico:** Gestión del modelo (añadir, eliminar, desplazar, rotar, escalar...) a nivel de vértice, arista y cara poligonal.
- **NURBS (Non Uniform Rational B-Splines):** Mediante curvas de aproximación, editando los puntos de control y los vectores de tensión, se define el contorno de la superficie que se generará en una etapa posterior. Las superficies NURBS presentan un aspecto muy suave, aunque son difíciles de manejar.
- **Operadores Booleandos:** Diferencias, Uniones e Intersecciones. Muy empleado en herramientas que trabajan con Geometía Sólida Constructiva (como herramientas CAD).
- **Extrusiones y Barridos:** A partir de un eje o camino de extrusión, se repite la forma a extruir para obtener una superficie.
- **Superficies de Revolución:** A partir de un eje de revolución y un contorno, se genera una superficie. Similar al modelado mediante un torno.
- **Superficies de Subdivisión:** Permiten un modelado suave, pero trabajando con formas poligonales. La malla original se subdivide para obtener la superficie límite, que será la que finalmente se represente.
- **Metasuperficies:** Algunos programas los denominan “metaballs” o “blobby surfaces”. Calculan la superficie limitada por un campo de fuerza.
- **Modelado procedural:** O calculado mediante scripts. Dentro de esta categoría están generadores de pelo, hierba, fluidos, árboles... Se basan en el uso de simuladores físicos y geometría fractal.

Debido a que la fase de modelado suele ser muy costosa, existen técnicas en investigación como el modelado basado en trazos 2D (por ejemplo, un programa Freeware llamado Teddy). En grandes proyectos se emplean escáner 3D (láser, basados en patrones de luz o de contacto), que generan desde modelos físicos una malla tridimensional.

c) Materiales y Texturas

Mediante los materiales, aplicamos propiedades básicas de reflexión de la luz, color, transparencia a las superficies de nuestros modelos. Existen modificadores de muchos tipos; rugosidad, brillo, desplazamiento, color, etc... Podemos ver la fase de aplicación de un material a un modelo como la asignación de la “piel” de este modelo. Una malla poligonal puede tener varios materiales asignados (en modo multimaterial).

Los materiales se aplican en capas, pudiendo modificar una o más propiedades en cada capa.

Se pueden emplear texturas de imagen en la definición de las propiedades del material. Existen diversos modos de proyección de las texturas (esférico, cilíndrico, cúbico, plano...). Un modo de ajuste muy preciso es el mapeado mediante coordenadas paramétricas UV (*UV Mapping*). Este tipo de mapeado es muy utilizado en aplicaciones interactivas (videojuegos), o en texturas asignadas tras un renderizado basado en Radiosidad.

d) Iluminación

En la búsqueda de la generación de imagen fotorrealista un punto clave es la simulación de la luz. La simulación siguiendo las leyes físicas de toda la luz dispersa en un modelo 3D sintético se denomina **iluminación global** (*global illumination*). El objetivo de las técnicas de iluminación global es simular todas las reflexiones de la luz y así obtener en cada punto del modelo el valor de intensidad de luz que vendrá determinado por la interacción de la luz con el resto de elementos de la escena. De esta forma, el objetivo del algoritmo de iluminación global es calcular la interacción de todas las fuentes de luz con los elementos que forman la escena. Estas técnicas de iluminación global son necesarias para calcular la iluminación indirecta; sin esta iluminación el modelo parece artificial y sintético.

La fase de iluminación está muy relacionada con la fase de renderizado. Debemos tener en cuenta la técnica de render a utilizar a la hora de iluminar la escena.

La **iluminación basada en imágenes** utiliza mapas de alto rango dinámico (HDRI) para capturar la iluminación real y asignarla al mundo virtual. Este tipo de técnicas, de utilización muy reciente, permiten obtener resultados de alto realismo sin incrementar el tiempo de generación de imagen 2D.

e) Animación

Tanto las herramientas de animación 2D como las 3D suelen emplear **curvas de interpolación** (en Blender se denominan curvas IPO) para calcular la posición del parámetro a animar (ya sea posición de un objeto, propiedades de material, etc...) entre frames clave. De esta forma, el usuario de la aplicación únicamente debe identificar esos fotogramas clave, establecer los valores de las propiedades en esos puntos y el software se encarga de calcular los valores intermedios.

En animación de personajes suelen emplearse **esqueletos** internos de animación. Así, el animador establece las rotaciones de estos huesos y el programa calcula la deformación de debe aplicar a la superficie exterior.

Para indicar la posición de los huesos del esqueleto, en la mayoría de las aplicaciones 3D, el animador cuenta con módulos de cálculo automático de la **cinemática inversa** (normalmente, implementados como aproximaciones iterativas, empleando el algoritmo CCD). De esta forma, el animador sólo debe ocuparse de indicar la posición de los huesos finales de la jerarquía, y el programa calcula la rotación de los huesos intermedios.

Si el presupuesto lo permite, pueden utilizarse sistemas de **captura del movimiento**. Existen principalmente dos tipos de sistemas de captura del movimiento; los basados en campos magnéticos y los ópticos. Los primeros funcionan mejor en aplicaciones interactivas, y los ópticos permiten utilizar un mayor número de marcas y, por lo general, ofrecen mayor precisión en la toma de datos (aunque requieren postprocesamiento para limpiar los valores calculados).

Si el proyecto lo requiere, pueden emplearse herramientas para la composición de imagen real con imagen virtual. Una aplicación de **Tracking de cámara** es imprescindible cuando se necesita componer una animación 3D con vide real. Dentro de estas herramientas, se puede encontrar el programa *Voodoo Camera Tracker* distribuido bajo licencia *Freeware*.

f) Render

En este paso se realiza el cálculo de la imagen 2D (de tipo mapa de bits, o imagen *Raster*) correspondiente a la escena que hemos definido. El motor de Render tiene en cuenta todos los parámetros definidos en las etapas anteriores, y trata de realizar una simulación física (más o menos aproximada, según el método de renderizado a utilizar) de la interacción de la luz en la escena. Existen varios métodos de render, siendo norma habitual que a mayor realismo, mayor tiempo de cómputo. El tiempo de render es un parámetro importantísimo a tener en cuenta a la hora de afrontar proyectos complejos. A continuación se detallan algunos métodos de render y sus propiedades:

- **Scanline:** Es un método de iluminación local. No permite calcular reflexiones y refracciones de luz (salvo empleando aproximaciones basadas en mapas de entorno y Transparencia en *Zbuffer* respectivamente).
- **Raytracing clásico:** Es un método de iluminación local. No simula correctamente sombras difusas. Debido al nombre de su inventor, se denomina también Raytracer tipo Whitted.
- **Radiosidad:** Simula la transferencia de energía entre superficies emisoras. Simula correctamente las reflexiones en superficies difusas, pero sólo es abordable utilizándolo con geometrías sencillas. La independencia de cálculo respecto del punto de vista lo hace muy interesante en visitas virtuales interactivas (donde suele cambiar únicamente el punto de vista de la cámara) y videojuegos (Quake II utilizaba este método para el sombreado de los escenarios).
- **PathTracing:** Emplea métodos de MonteCarlo para calcular "todos" los posibles caminos de luz en superficies difusas. En realidad se traza un rayo aleatoriamente (dentro del dominio de la función) para estimar la luz que viene de todas las direcciones.
- **PathTracing Bidireccional:** Además de ver los caminos de luz desde el observador, también calcula los caminos de luz desde las fuentes. Ofrece mejores resultados, aunque en escenas con iluminación no demasiado complejas, es más lento que el PathTracing.

g) Postproducción

Algunos efectos (como simulación de profundidad de campo, motion Blur, etc...) es menos costoso generarlos independientemente y componerlos en la etapa de Postproducción empleando capas.

En esta fase se suelen retocar individualmente los fotogramas que formarán nuestra animación, ajustando niveles, brillo, contraste, etc... En este punto suelen incorporarse también efectos de sistemas de partículas (como nieve), iluminación (flares), etc...

La correcta planificación de la etapa de postproducción puede disminuir notablemente el tiempo global a emplear en el proyecto. Un ejemplo típico es la generación de efectos de desenfoque de cámara, muy costosos de simular mediante renderizado y fácilmente realizables mediante una herramienta de desenfoque en un programa de retoque.