Práctica 2: Trazador de rayos

(Asignación 6 de Marzo; Entrega 27 de Abril a las 15:00)

Grupos y Entregas

La práctica se puede realizar en parejas o de forma individual. Una pareja puede estar formada por una persona que sólo siga la parte obligatoria de la asignatura y una persona que siga también la parte opcional, y a cada persona se le evaluará de los módulos correspondientes.

La entrega constará de:

- Ejecutable y ficheros de escena (y dlls adicionales si se utilizan librerías externas).
- Código.
- Pequeña memoria con: opciones del código, lista de funcionalidades implementadas, imágenes representativas de los resultados obtenidos. No se busca una memoria extensa que explique el algoritmo de traza de rayos, sino una memoria escueta que indique de forma rápida qué se ha implementado.
- Si se usa código externo, ideas de otras referencias, ayuda de compañeros... se ha de citar en la memoria. En caso contrario, el trabajo se considerará como copia.

La entrega se realizará por Campus Virtual, y se habilitará una entrada para ello. Si la práctica se ha realizado en pareja, sólo una persona deberá subir el material. La fecha límite para la entrega es el 27 de Abril a las 15:00.

Especificaciones Generales:

Se proporciona un proyecto sobre MS VisualStudio 2010 (se puede hacer update hasta 2013) que servirá de esqueleto para la práctica. En este proyecto se programarán los siguientes bloques de funcionalidades:

<u>Bloque 1 (obligatorio para todos):</u>

- Algoritmo de ray casting e iluminación directa. Es decir, replicar el comportamiento de OpenGL básico.
- Incorporación de sombras.
- Algoritmo de traza de rayos recursivo con 4 rebotes.
- Geometría: esferas y triángulos individuales.
- Materiales: lambertianos, con reflexión tipo Phong, y mapeado de texturas (sin filtrado).
- Luces puntuales.

<u>Bloque 2 (obligatorio para quienes sigan la asignatura completa, opcional para el resto):</u>

- Modelos externos descritos como mallas de triángulos.
- Supermuestreo (supersampling) 4x en el plano imagen para antialiasing.
- Iluminación global por Monte Carlo con 32 muestras por rebote.
- Vídeo walk-through de una escena con camino de cámara pre-grabado (con y sin Monte Carlo).

Bloque 3 (Al menos 3 extensiones obligatorias para quienes sigan la asignatura completa, opcional para el resto):

- Refracción.
- Filtrado de texturas.
- Mapeado de texturas avanzado (normal mapping, displacement mapping, etc.)
- Simulación de lentes y/o efecto de campo de profundidad.
- Motion blur.
- Luces de área.
- Iluminación global por traza de caminos.
- Estructuras de aceleración (cajas/esferas envolventes, Kd-trees...).
- Implementación (de una o varias funcionalidades) en CUDA/GPU.
- Medios participativos
- ...

Para la evaluación, se tendrá en cuenta el haber completado (y con qué nivel de calidad) los apartados obligatorios. Los apartados opcionales completados servirán para subir nota (Sin límite. Si alguien completa los 3 Bloques, tiene un 10 en la asignatura).

Esta práctica está basada casi en su totalidad en la práctica 3 de la asignatura 15-462 de la Universidad Carnegie Mellon, dirigida en su momento por Alexei Efros y Nancy Pollard.

Código y Parámetros:

El punto de entrada del código a implementar es la función:

```
Vector RayTrace::CalculatePixel(int screenX, int screenY)
```

Se encuentra en el fichero RayTrace.cpp.

Para ejecutar el código, hay que pasar un fichero de escena en formato .xml, por ejemplo el proporcionado como ejemplo, test.xml.

Información a tener en cuenta:

- (screenX, screenY) indican las coordenadas de pantalla de un pixel concreto. Las coordenadas comienzan en la esquina inferior izquierda (0,0) y la recorren hasta llegar a la esquina superior derecha (Scene::WINDOW_WIDTH-1, Scene::WINDOW_HEIGHT-1).

- Para trazar un rayo, primeramente se ha de formular la ecuación del rayo, que requiere el conocimiento de los parámetros de cámara. Dichos parámetros se modifican mediante teclado (letras o flechas), en las funciones globales processkeys() y processarrowkeys(), en un objeto de tipo Camera. Atención: hacen referencia a parámetros de cámara de tipo OpenGL.
- Una vez conocida la ecuación del rayo, se calculará la intersección con la escena, se aplicará el modelo de iluminación, y se devolverá el color del rayo. La escena está almacenada en el objeto Scene m_Scene de la clase RayTrace. Este objeto se inicializa a partir del fichero .xml al comienzo de la aplicación.
- La ejecución de las opciones de supermuestreo y MonteCarlo se determina por el menú, y viene indicada en las variables Scene::supersample y Scene::montecarlo.

Depuración:

Para depurar el código, conviene seguir los siguientes consejos:

- Se puede determinar como color resultado cualquier variable que se calcule en el trazador. Es suficiente con acotar el valor al rango [0,1].
- Ante la presencia de errores, conviene reducir la complejidad de la escena (cámara en el origen, sólo un triángulo y en posiciones sencillas, etc.).
- Finalmente, si no se consigue encontrar el error habrá que mirar lo que sucede en un pixel concreto.