Examen de laboratorio de IDI 2017-2018, Q1 Tiempo: 1h 55'

Lee detenidamente las instrucciones y el enunciado antes de empezar a trabajar!

Instrucciones

1. Puedes usar el código que has elaborado en las clases de laboratorio y que tengas en tu cuenta, pero **solamente el código que hayas generado tu**. No puedes usar código que otros estudiantes hayan compartido contingo ni que tu hayas compartido con otros. De lo contrario se considerará copia.

2. Partirás del código contenido en el fichero `examen.tgz` (adjunto a esta práctica). Tienes que descomprimirlo en un directorio tuyo. Se creará el subdirectorio `examen-1718Q1` donde encontrarás todos los ficheros con los que tienes que trabajar. Los ejercicios a resolver sólo requieren cambios en la clase `MyGLWidget`, en los shaders y en el fichero `MyForm.ui` usando el `designer`. **No tienes que modificar ningún otro fichero de los que te proporcionan**.

3. **Si el código que entregas no compila o da error de ejecución, la evaluación será un cero**, sin excepciones.

4. Para hacer la entrega tienes que generar un archivo tar que incluya todo el código de tu examen y que se llame `<nombre-usuario>.tgz`, substituyendo `<nombre-usuario>` por tu nombre de usuario. Por ejemplo, el estudiante Pompeu Fabra (desde el directorio `examen-1718Q1`):

   ```
   make distclean
   tar zcvf pompeu.fabra.tgz *
   ```

   Es importante el '`make distclean`' para borrar los archivos binarios generados; que el nombre de usuario sea el tuyo y que tenga el sufijo `.tgz`.

5. Una vez hecho esto, en tu directorio `examen-1718Q1` tendrás el archivo `<nombre-usuario>.tgz` que es lo que tienes que entregar. **Comprueba** que todo es correcto descomprimiéndolo **en un directorio vacío** i mirando que el código compila (haciendo `qmake-qt5; make`) y ejecuta correctamente.

6. Finalmente, entrega el fichero en `https://examens.fib.upc.edu`.

Nota: Recuerda que si abres el fichero `~/examen/assig/idi/man_3.3/index.html` desde el navegador tendrás acceso al manual de OpenGL 3.3, y que en `~/examen/assig/idi/glm/doc/api/index.html` tienes el manual de la librería `glm`. También tienes, como ya sabes, el `assistant-qt5` para dudas de Qt.

Enunciado

Tal y como está, el código que te proporcionamos pinta sobre el plano XZ un suelo centrado en el origen, de 20×20 unidades, con un Legoman de una unidad de altura y el centro de la base de la caja que lo contiene situado en el origen de coordenadas. Mira la imagen del archivo `EscIni.png`. La escena se representa con una cámara completamente arbitraria de la que solamente puede modificarse interactivamente el ángulo ψ.

Fíjate que se ha definido un método `createBuffers` para cada modelo, con el propósito de simplificar la lectura del código y su posterior utilización. Este método tiene inicializados todos los datos de material y normales necesarios para poder implementar el cálculo de la iluminación. También se proporcionan las funciones `Lambert` y `Phong` que se encuentran en el `Vertex Shader`.

Este examen puntua sobre 12. Se pueden conseguir hasta 12 puntos si se realiza todo correctamente.

Se dará importancia al diseño y la usabilidad de la interfície que se pide (ejercicios 4, 6 y 7).

1. *(2 puntos)* Modifica la escena dada de manera que el Legoman sea substituido por 4 Patricios (modelo `Patricio.obj`) uniformemente escalados de altura 5, situados tal y como muestra la imagen que tenéis en el fichero `EscSol1.png`. Concretamente: el primer Patricio (el que está debajo) tendrá el centro de su base en el punto (0, 0, 0), el segundo Patricio tendrá el centro de su base en el punto (-2.25, 2.75, 0), el tercero tendrá el centro de su base en el punto (2.25, 2.75, 0) y el cuarto y último tendrá el centro de su base en el punto (0, 5.5, 0). Todos ellos estarán mirando en dirección Z+. Recuerda que el Patricio inicialmente mira hacia las Z+.

   También debes modificar el material del suelo para que pase a ser un material de color magenta y brillante.

2. *(2 puntos)* Para poder inspeccionar la escena descrita, programa una cámara en tercera persona y óptica perspectiva que inicialmente permita ver la escena centrada, completa, sin deformar y ocupando el área máxima del viewport (siendo el viewport toda la ventana gráfica). En caso de redimensionar la ventana (`resize`) la escena no se deformará ni se recortará. La cámara permitirá la inspección de la escena mediante rotaciones de los ángulos de Euler (ángulos ψ y θ), es decir, el usuario tiene que poder modificar dichos ángulos utilizando el ratón como se ha visto en el laboratorio. Inicialmente la cámara debe tener ángulos $\psi = M\_PI/4.0$ i $\theta = 0$.

   Una imagen ilustrativa de la solución a estos dos primeros ejercicios puedes verla en el archivo `EscSol1.png`.

3. *(1.5 puntos)* En el **Fragment Shader**, añade el cálculo de la iluminación de la escena usando el modelo de Phong con un foco de cámara de luz blanca situado siempre exactament en la posición de la cámara.

4. *(1.5 puntos)* Programa que el grupo de los 4 Patricios gire respecto del eje Y de la escena. El ángulo de giro deberá ser controlado por un elemento (widget) que deberás añadir a la interfaz y que debería ser el más adecuado para controlar este ángulo. Los Patricios deben girar todos en bloque, todo el pilar a la vez, mantniendo su posición relativa entre ellos.

   El giro de los Patricios debe funcionar independientemente de la cámara que esté activa (ver ejercicio 5).

5. *(1 punto)* Añade una segunda cámara fija en la posición (10, 5.25, -10) de la escena y enfocando hacia el centro de la escena. La óptica de la cámara será perspectiva con ángulo de apertura de `M_PI/2.0` radianes (90 grados) con posición y orientación fijas. Los valores de `Znear` y `Zfar` permitirán ver todo lo que pueda verse de la escena desde la posición de la cámara y en la dirección antes indicada. Caso de redimensionado del viewport, la cámara no deformará la escena.

   Esta nueva cámara debe poder activarse/desactivarse mediante la tecla 'C'. Cuando se desactiva esta nueva cámara y se pasa a la cámara del ejercicio 2 deben recuperarse los últimos valores que ésta había tenido (ángulos ψ y θ).

   Para la posición y orientación de esta segunda cámara puedes utilizar, si quieres, la función `lookAt` de la libreria `glm`.

6. *(1.5 puntos)* Añade el elemento de interfaz adecuado para que el usuario pueda decidir en todo momento qué cámara quiere tener activa, la del ejercicio 5 o la del ejercicio 2.

   Este elemento de interfaz debe ir correctamente coordinado con el uso de la tecla 'C' (que también se usa para hacer el cambio entre cámaras).

7. *(2 puntos)* En el ejercicio 6 de la sesión 2.4 del laboratorio (Bloc-2) se pedía lo siguiente:

   > Implementa una clase derivada de *QLabel* (MyLabel) como clase propia que permita mostrar mediante el color de su *background* el color representado por 3 *Spinbox* que definen los valores de R, G y B del color mostrado.

   Aprovecha la misma clase MyLabel que hiciste para este ejercicio y úsala en este caso para decidir el color del foco de luz.