Facultad de Ingeniería
Abog. Lorenzo Zacarías c/ Ruta 1 Km 3

Proyecto #1

Escena 3D
Gráficos por computadoras

OBJETIVO

Aprender a modelar una escena en 3D usando transformaciones geométricas para organizar los objetos en la escena. Jugar con OpenGL

INSTRUCCIONES

Su tarea consiste en completar los ejercicios del Capítulo 6 del libro “Foundations of 3D Computer Graphics” de Steven J. Gortler.

A continuación hago una copia exacta de los ejercicios tal cual como estan en el libro, para que no haya dudas de cuales ejercicios se trata.

Ejercicios:

Ex. 14— In this chapter, we have explained the basics of an OpenGL 3D “Hello World” program. At the book’s website, we have some basic starter code, including the necessary Cvec and Matrix4 classes. Download and compile this program.

Ex. 15— Add code to allow the user to move the blue cube with respect to the auxiliary coordinate frame of Equation (5.6). The mouse motion should be interpreted as follows:

•Left button: moving the mouse right/left rotates around the y-direction.

•Left button: moving the mouse up/down rotates around the x-direction.

•Right button: moving the mouse right/left translates in the x-direction.

•Right button: moving the mouse up/down translates in the y-direction.

•Middle button (or both left and right buttons down): moving the mouse up/down translates in the z-direction.

Ex. 16— Use the 'o' key to let the user toggle between the blue cube and red cube. This lets the user manipulate either object.

Ex. 17— Use the 'o' key to let the user toggle between the blue cube, the red cube and the eye to let the user manipulate either object, or the eye. For the eye motion, apply all transformations with respect to $^{\sim}e_t$ (ego motion), and invert the signs of the translations and rotations.

Ex. 18— Replace each of the cubes with a tessellated sphere. To tessellate a sphere, parametrize each point on the unit sphere with angular coordinates $\theta \in [0..\pi]$ and $\varphi \in [0..2\pi]$. Define the object coordinates of each point using

$x = \sin\theta \cos\varphi$

$y = \sin\theta \sin\varphi$

$z = \cos\theta$

Ex. 19— (Lengthy) Replace each of the cubes with a simple robot. Each robot will have movable joints at the shoulders, elbows, hips, and knees. Use the numeral keys to let the user pick the relevant joint. Let the shoulder and hip have full 3D rotational degrees of freedom. Let the elbow have just two rotational degrees of freedom, and the knee just one rotational degree of freedom. For joints with 1 or 2 degrees of freedom, you should store these as Euler angles corresponding to rotations about the appropriate (e.g., x or z) axes. The body parts should be drawn as non-uniformly scaled cubes or spheres.

Ex. 20— The user should be allowed to press 'p' and then use the mouse to left click

on a desired robot part. The determination of which part the user clicked on is called “picking”. After picking a part, the user should then be allowed to use the mouse to rotate the part about its joint.

To do picking we follow these steps: when ’p’ is pressed, the next left mouse click will trigger the following. The current scene is rendered, but this time, instead of drawing the usual colors into the color back buffer, you will assign each part to be drawn with its own unique color into the buffer. Since each part’s color is unique, it will serve as an ID for the part. Drawing this ID-tagged scene will involve using a fragment shader that does not do lighting calculations but instead directly returns its input color. After rendering the scene, call glFlush(), and then read the ID from the back buffer using a call to glReadPixels to determine which part was selected by the mouse. To set the backbuffer as the read target, use glReadBuffer(GL BACK) at some point before the glReadPixels call. Also, in your initialization code, you should call glPixelStorei(GL UNPACK ALIGNMENT, 1), so you can correctly access the pixel data.

•If the mouse click was over a robots part, then the chosen robot part is selected.

•If the mouse click was over a robots main body, the entire robot is selected.

•If the mouse click was not over any robot, then the eye is selected for manipulation.

Once picking is done, Subsequent mouse motion with the mouse button depressed should manipulate the selected object or joint.

PUNTAJE EXTRA

1. En el ejercicio 19, solo la parte de mostrar los Robots es obligatoria, la parte de Transformacion en las uniones ya es Extra (Opcional).
2. Todo el ejercicio 20 es Opcional.

FORMA DE ENTREGA

1. Un informe explicando lo hecho y los problemas no resueltos
2. Una carpeta que contenga capturas de pantalla de los ejercicios resueltos.
3. Una **readme.txt** conteniendo instrucciones necesarias para ejecutar el proyecto.
4. Subir un archivo comprimido con todo el código fuente y archivos requeridos en la tarea del classroom.