Computación Gráfica - TP: Aplcación de Texturas

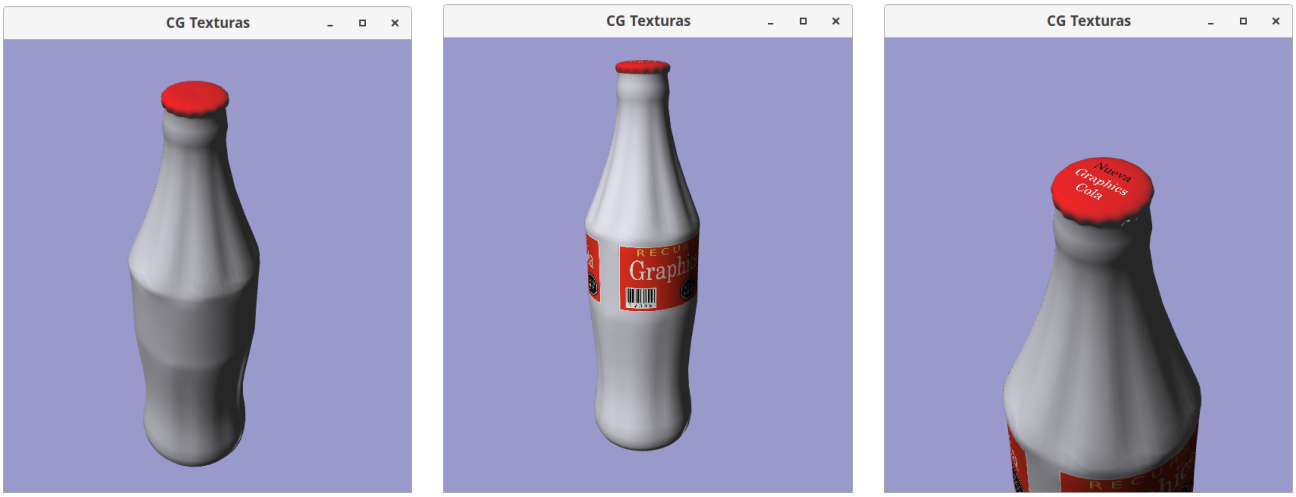
1. Resumen de tareas

1. Implementar la generación automática de coordenadas de textura utilizando en cada caso el método que considere más adecuado para los modelos del ejemplo: una botella y su tapa (funciones `generateTextureCoordinatesFo` y `generateTextureCoordinatesForLid` en *main.cpp*) .
2. Analizar los problemas encontrados en el resultado y proponer (no es necesario implementar) posibles soluciones.
3. Analizar cómo se combinan la información que se obtiene de la textura con el cálculo de iluminación.
4. Opcional: implementar un shader que permita *visualizar* las coordenadas de textura.

2. Consigna detallada

2.1. Generación de las coordenadas de textura

El código inicial carga en memoria y renderiza dos *modelos* (dos mallas en dos instancias de `Model`), una correspondiente a una botella, y otra correspondiente a la tapa de la misma. En los datos de entrada (el archivo *bottle.obj*) se encuentran las coordenadas de los vértices, las normales de los mismos y las conectividades (los triángulos); pero **la información de entrada no contiene coordenadas de textura para los vértices. El alumno debe generarlas (es decir, calcularlas con algún método automático en función de las coordenadas de los vértices)**, para poder aplicar las texturas que se cargan desde los archivos *label.png* y *lid.png*. El resultado esperado es el siguiente:



Las funciones `generateTextureCoordinatesForBottle` y `generateTextureCoordinatesForLid` (las que deberá completar, definidas ambas en *main.cpp*), reciben como argumento de entrada un vector con los nodos (coordenadas) de la malla (`std::vector<glm::vec3>`), y deben retornar un nuevo vector del mismo tamaño con las coordenadas de textura para dichos nodos (`std::vector<glm::vec2>`). Cada elemento del vector de salida se corresponde con el elemento del vector de entrada de la misma posición (índice).

Si realiza una primera implementación que no resuelve correctamente el problema, puede pasar al paso 4 (Visualización de las Coordenadas de Textura) para lograr así un mejor mecanismo de depuración que le permita entender los problemas.

2.2. Problema esperado

Es probable que luego de implementar las funciones indicadas en el punto anterior, se encuentre con el siguiente problema:

En una parte de la botella se encuentra repetida y aplastada toda la etiqueta. **Analice el problema para entender la causa del mismo y proponga una posible solución (no debe implementarla**, la solución requerirá trabajo adicional fuera de las dos funciones modificadas en el primer paso).

2.3. Iluminación y Textura

Analice cómo se combina la información de color que provee la textura con la información del material (constantes para el modelo de *Phong*) y el cálculo de iluminación. Deberá primero encontrar en qué *lugar* del código se produce dicha combinación. Ayuda: piense en qué *lugar* del *pipeline* debería resolverse estre problema.

2.4. Visualización de las Coordenadas de Textura

El práctico cuenta con una opción (la tecla *C* o el check ”Use shader coords” en la GUI) para utilizar un shader alternativo. Inicialmente este shader tendrá el mismo código que el shader principal que se utilizar para aplicar la textura. Tomando esto como base, **modifique el shader para lograr ”visualizar” de alguna forma las coordenadas de textura asignadas a cada fragmento**.

Ayuda: recuerde que el valor a utilizar siempre estará entre 0 y 1, pero los valores *s* y *t* que recibe un fragmento podrían no estarlo (se ajustan mediante los mecanismos de *clamp* o *repeat* al consultar la textura).