Universidad Central de Venezuela
Facultad de Ciencias
Postgrado en Ciencias de la Computación
Introducción a la Computación Gráfica

Caracas, 23 de Diciembre de 2025.

Proyecto # 2

Se requiere que usted realice una aplicación utilizando OpenGL/WebGL y GLSL, bien sea en c++ o Javascript con las siguientes características:

- Inicialización (3.5 puntos)
    a. Cargar OBJ. En c++, se puede utilizar algún "tiny open file dialog" para seleccionar el archivo a abrir. Se asume que el .MTL está en el mismo directorio donde se encuentra el .OBJ. Si no se encuentra el MTL, asignar un color gris 0.7, 0.7, 0.7 como color por defecto a todos los sub mallados del OBJ (diffuse color), e indicar el problema en un simple mensaje. Asuman que el color del objeto es el que viene en el campo Kd del MTL. El resto de los atributos de color (Ka, map_Kd, etc.) los pueden leer y cargar, pero ignorar en este proyecto en términos de rendering. Cada OBJ tiene sub mallados, y cada sub mallado podría tener un material diferente (2.5 puntos). Más información sobre el formato, leer:
    https://en.wikipedia.org/wiki/Wavefront_.obj_file
    b. Normalizar el OBJ de manera tal que entre dentro de un cubo unitario, pero manteniendo sus proporciones. Para ello, recomiendo inicializar los atributos de "center" (vec3), "scale_factor" (vec3), de manera de durante el rendering se realicen esas transformaciones iniciales, y el objeto esté centrado en 0 y con escala unitaria (1 punto): scale(scale_factor) * translate(-center). El objeto debe ser luego trasladado a z = –3, y el ojo ubicado en 0 viendo hacia –z.

- Edición general del objeto (10 puntos):
    a. Seleccionar el "sub mallado actual" vía picking (1 punto). Para ello, renderice en el back buffer cada sub mallado con un color único, y utilizando glReadPixels determinar que objeto fue seleccionado.
    b. Si el objeto no viene con sus normales por vértice, aproximar la normal de cada vértice como el promedio de las normales de sus triángulos. Este algoritmo debe ser O(n) (1 punto)
    c. Trasladar el objeto completo, si no hay ningún sub mallado seleccionado (0.75 puntos). Recomiendo utilizar el ratón, y/o edit boxes de interfaz.
    d. Cambiar la escala del objeto, modificando scale_factor, independientemente por cada eje. Recuerde que scale_factor tiene

unas escalas iniciales que “normalizaron” el objeto. El usuario puede cambiar cada una de estas escalas iniciales a placer. (1 punto)

e. Rotar el objeto sobre su propio centro con el ratón. Note que el centro del objeto puede cambiar, puesto que los sub mallados pueden haber sido trasladados. La rotación debe ser en la dirección de movimiento del ratón. Debe mantener traza de todas las rotaciones realizadas. Se recomienda utilizar quaterniones o de lo contrario, llevar traza de la acumulación de las matrices de rotaciones (2 puntos).
f. Centrar el objeto. Dado que los sub mallados pueden haber cambiado de posición, un botón de “center” nunca estará demás. El centro del objeto, y la escala a cubo unitario deben recalcularse según el bounding box actual, y colocar el ojo en 0 viendo hacia –z, mientras el objeto se traslada a z = –3 (1 punto).
g. El usuario puede moverse hacia adelante y hacia atrás desde su posición, en la dirección del vector de vista (vista = destino - posición) con las teclas UP y DOWN del teclado (0.75 puntos)
h. El usuario puede rotar su vista con las teclas LEFT y RIGHT (0.5 puntos), pero también puede utilizar el ratón (1 punto) para cambiar la dirección de visualización de forma “arbitraria”. Se recomienda llevar traza de los vectores front_vector, left_vector o right_vector, y up_vector como en los juegos FPS. Eso facilitará enormemente la creación de la matriz de vista (lookat).
i. El objeto 3D modificado puede ser guardado y cargado posteriormente, también en formato OBJ + MTL (1 punto). Note que el objeto y/o sub mallados fueron trasladados, el objeto fue rotado y/o escalado, el color de cada sub mallado pudo haberse cambiado. Todas las transformaciones debe ser aplicadas sobre los vértices, puesto que el OBJ no guarda matrices de transformación.

- Opciones de interfaz globales (4.5 puntos)
  a. Ver/Ocultar vértices de los sub mallados (0.25 puntos), cambiar color RGB de los vértices Kd (0.25 puntos), y tamaño de vértices (0.25 puntos). Usar glPolygonOffset o equivalente para evitar z-fighting (0.25 puntos)
  b. Ver/Ocular wireframe o alambrado (0.25 puntos), cambiar color de alambrado (0.25 puntos), usar glPolygonOffset o equivalente para evitar z-fighting (0.25 puntos)
  c. Ver/Ocultar relleno de triángulos (0.25 puntos)
  d. Ver/Ocultar la normal por vértice (0.5 puntos), cambiar color RGB de las normales (0.25 puntos). Usar un % de la longitud de la diagonal de bounding box en coordenadas de mundo para definir la longitud de las normales (0.5 puntos)
  e. Habilitar/deshabilitar z-buffer o depth test (0.25 puntos)
  f. Habilitar/deshabilitar back-face culling (0.25 puntos)

g. Mostrar/ocultar frames per second como promedio de los últimos 5 segundos (0.25 puntos)
h. Habilitar/deshabilitar antialiasing de líneas (0.25 puntos)
i. Cambiar el color del background (0.25 puntos), el cual es negro por defecto.

- Sobre el sub mallado seleccionado vía picking, permitir (2 puntos):
  a. Cambiar su color RGB (Kd) (0.25 puntos)
  b. Trasladar el sub mallado seleccionado (0.5 puntos)
  c. Eliminar sub mallado (0.5 puntos)
  d. Mostrar bounding box únicamente para el sub mallado seleccionado (0.25 puntos). Evitar z-fighting del bounding box con el sub mallado (0.25 puntos). Permitir cambiar el color de bounding box (0.25 puntos)

Entrega:

- Se debe entregar (01 de Febrero de 2026): enviar LINK de dropbox del .zip de toda la carpeta a mi correo recs34@gmail.com (únicamente el LINK de dropbox), para que no rebote el correo. Indicar nombre en dicho correo, colocando como subject “Proyecto 2 – Gráfica <nombre>”.
- Se penaliza 1 punto por día de retardo en la entrega.

ÉXITOS

R.C./rc