Gráficas por Computadora

Profesor: Mario Martínez Molina

Tarea 2: Renderizado con OpenGL

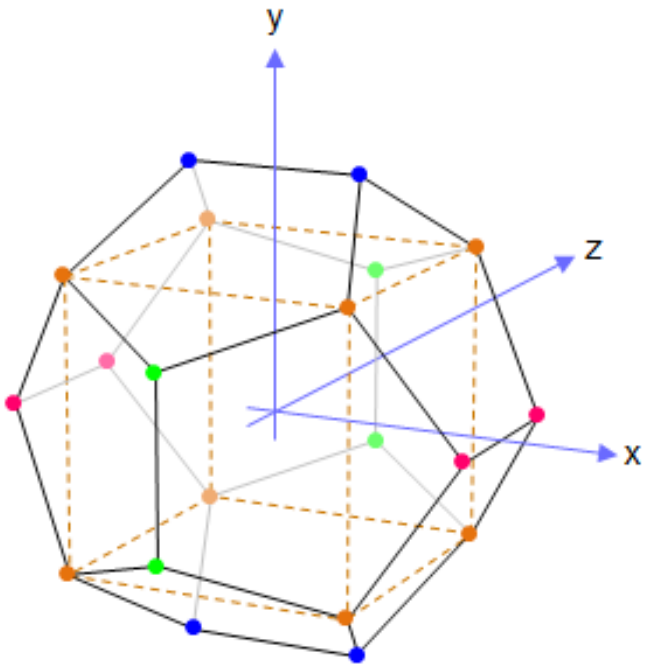
1. Cree una aplicación con OpenGL que realice el renderizado en pantalla de cada uno de los tetrominos del juego “tetris”. Utilice una variable uniforme llamada `color` y un solo shader de fragmentos para renderizar cada modelo con un color distinto.

2. Repita el ejercicio anterior usando un shader de fragmentos distinto por cada modelo renderizado.

3. Las coordenadas cartesianas de los vértices de un dodecahedro centrado en el origen están dadas como sigue:
   - $(\pm 1, \pm 1, \pm 1)$. Estos vértices definen un cubo centrado en el origen (vértices naranja en la figura).
   - $(0, \pm \frac{1}{\phi}, \pm \phi)$. Estos vértices forman un rectángulo sobre el plano yz (vértices verdes en la figura).
   - $(\pm \frac{1}{\phi}, \pm \phi, 0)$. Estos vértices definen un rectángulo sobre el plano xy (vértices azules en la figura).
   - $(\pm \phi, 0, \pm \frac{1}{\phi})$. Estos vértices forman un rectángulo sobre el plano xz (vertices rojos en la figura).

   donde $\phi \approx 1.618$ es el número áureo o razón dorada. Escriba un programa que use la especificación 3.3 de OpenGL para renderizar este poliedro a través de la función `GLuint crearModelo(GLuint* vbo, GLuint* ebo)`, de acuerdo a las siguientes especificaciones:

   - Cada vértice del modelo contiene los siguiente atributos: un atributo de **posición** de tres dimensiones, y un atributo **color** de cuatro dimensiones.
   - La función debe realizar las siguientes tareas: crear e inicializar los arreglos de datos que serán usados para inicializar los objetos buffer para el renderizado, crear e inicializar los objetos buffer y objetos vertex array necesarios para renderizar el dodecahedro mediante renderizado indexado, y realizar la especificación de vértices a la tubería gráfica.



4. Había una vez un ratón que había pasado muchos días sin comer, esto debido a que el y toda su familia fueron ahuyentados de la casa donde vivían por un malvado gato. Un buen día mientras el ratón buscaba algo que comer pudo percibir un fuerte olor a queso, tras buscar un rato, el ratón encontró un pequeño trozo de este manjar. Después de comerlo, el ratón se dio cuenta que aún percibía el aroma a queso, rápidamente nuestro ratón pudo encontrar otro trozo más. !El ratón había encontrado un camino de queso!

   Escriba un programa usando la especificación 3.3 de OpenGL y el perfil core que renderice la historia anterior de acuerdo a las siguientes especificaciones:

   a) La posición del ratón está dada por el vector de posición $R=(-0.7,-0.8,0.0)$. Renderice al ratón mediante un triángulo con una escala de 0.05 y un color RGBA igual a $(0.5,0.25,0.25,1.0)$.

   b) Renderice cada trozo de queso mediante un cuadrado con escala de 0.05 y color RGBA igual a $(1.0,1.0,0.0,1.0)$. Las posiciones de cada trozo de queso están dadas por los siguientes vectores de posición:

      - $Q_1=(-0.5,-0.5,0.0)$
      - $Q_2=(-0.2,-0.4,0.0)$
      - $Q_3=(0.0,-0.1,0.0)$
      - $Q_4=(0.2,0.2,0.0)$
      - $Q_5=(0.0,0.5,0.0)$

c) Anime su escena de manera que la posición del ratón cambie cada segundo (use la función `glfwGetTime()` para calcular este intervalo de tiempo) a la posición de cada trozo de queso correspondiente.

La fecha de entrega límite para esta tarea es el miércoles 31 de enero de 2018.