Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Programación Optimizada para Videojuegos

Master Universitario en Desarrollo de Videojuegos

Primera actividad WebGL WebGlCube

Juan Siverio Rojas

La Laguna, 23 de noviembre de 2022

Contenido

1.1 Objetivo

Tras realizar el tutorial 1, que te permite renderizar un cubo en rotación, modifica el proyecto para que se pueda mostrar un color diferente en cada cara del cubo.

1.2 Abstract:

I modified the index buffer, vertext buffer and color buffer for to represent a cube of 24 vertices, split into groups of four for each face and associating each group with a unique color. Also, I modified the rotation angle at X, for can show all faces.

1.3 Proceso

En el proyecto original, se dispone de un cubo de ocho vértices y la primitiva utilizada ha sido el triángulo, dando lugar a doce fragmentos (triángulos) utilizados por el shader de pixels para dar color mediante interpolación. Los cuatro vértices correspondientes a la cara frontal son de color rojo y los cuatro vértices de la cara trasera son verdes, el resto de caras del cubo, en las que los fragmentos ya utilizan vértices tanto de la cara frontal como de la trasera, adoptan el color interpolado.

Para establecer la ubicación de cada vértice y los índices de las primitivas establecidas en el código original, he llevado esta información a la web Geogebra, la cual permite graficas en 3D basado en vértices. De esta forma, pude conocer exactamente que cara es cada cual, y que vértices están asociados a cada primitiva.(los vértices están enumerados desde el 0 al 7)

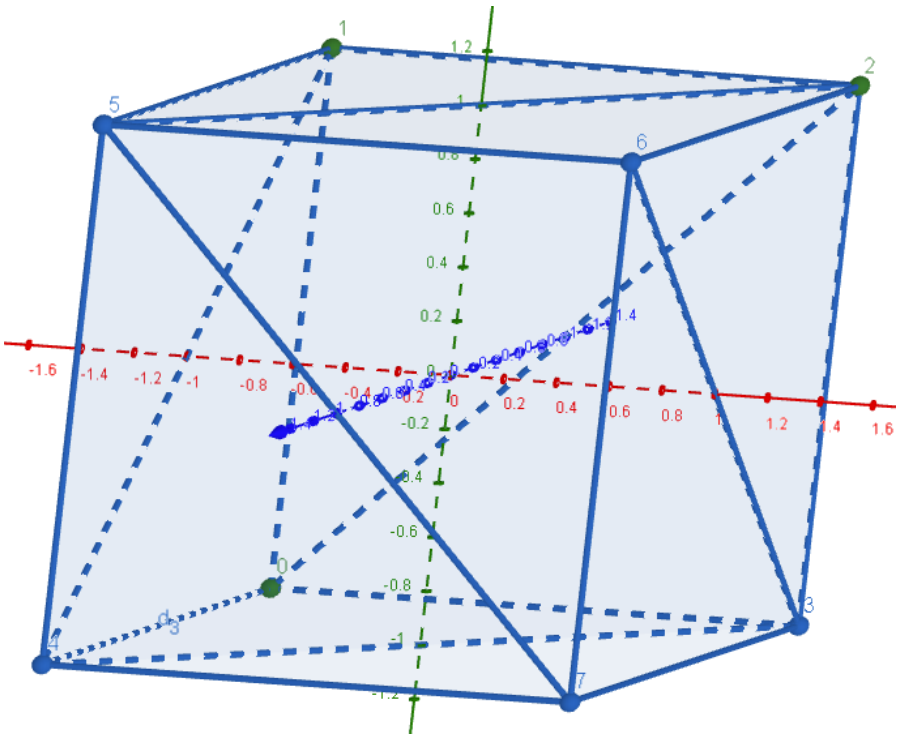

Ilustración 1: Representación 3D del modelo original

Pues, para conseguir que cada cara del cubo tenga un color diferente, he creado un cubo de 24 vértices, cuatro por cada cara, y asociándoles un mismo color a los cuatro vértices de una misma cara. Se modifica también el buffer de índices para garantizar que los fragmentos de una misma cara sean vértices que tienen el mismo color.

Para ver correctamente el efecto, además, se establece una rotación también en el eje X, debido a que con la configuración original, en la que solo se modifican las componentes Z e Y, las caras inferior (Bottom) y trasera (Back) no se logran mostrar. Esto se realiza en el método Update().

```
Vector3 axis = new Vector3(0.5f,0.5f,0.5f);
```

Ilustración 2: Cambio rotación

El código nuevo es simplemente sustituir buffer de vértices, el de color y el de los índices.

```
private static readonly float[] cubeVertices = {
      //Side Back
      -1.0f,-1.0f,-1.0f, //0
      -1.0f,1.0f,-1.0f, //1
      1.0f,1.0f,-1.0f,  //2
      1.0f,-1.0f,-1.0f, //3

      //Front
      -1.0f,-1.0f,1.0f, //4
      -1.0f,1.0f,1.0f,  //5
      1.0f,1.0f,1.0f,  //6
```

```
    1.0f,-1.0f,1.0f,  //7

    //Rigth
    1.0f,1.0f,1.0f,  //6 as 8
    1.0f,1.0f,-1.0f,  //2 as 9
    1.0f,-1.0f,1.0f,  //7 as 10
    1.0f,-1.0f,-1.0f, //3 as 11
    //Left
    -1.0f,1.0f,-1.0f, //1 as 12
    -1.0f,-1.0f,1.0f, //4 as 13
    -1.0f,1.0f,1.0f,  //5 as 14
    -1.0f,-1.0f,-1.0f, //0 as 15
    //Top
    -1.0f,1.0f,-1.0f, //1 as 16
    1.0f,1.0f,-1.0f,  //2 as 17
    -1.0f,1.0f,1.0f,  //5 as 18
    1.0f,1.0f,1.0f,  //6 as 19
    //Bottom
    -1.0f,-1.0f,-1.0f, //0 as 20
    1.0f,-1.0f,-1.0f, //3 as 21
    -1.0f,-1.0f,1.0f, //4 as 22
    1.0f,-1.0f,1.0f  //7 as 23

};

private static readonly int[] intCubeIndices =  {
    //Back
    2,1,0,
    3,2,0,
    //Front
    5,7,4,
    6,7,5,
    //Right
    8,9,11,
    10,8,11,
    //Left
    12,13,15,
    14,13,12,
    //Top
    17,18,16,
    17,19,18,
    //Bottom
    22,21,20,
    23,21,22
};
```

```
private float[] cubeColors= new [] {
    //Back Red
    1.0f,0.0f,0.0f,1.0f,
    1.0f,0.0f,0.0f,1.0f,
    1.0f,0.0f,0.0f,1.0f,
    1.0f,0.0f,0.0f,1.0f,
    //Front Green
    0.0f,1.0f,0.0f,1.0f,
    0.0f,1.0f,0.0f,1.0f,
    0.0f,1.0f,0.0f,1.0f,
    0.0f,1.0f,0.0f,1.0f,
    //Right Pink
    1.0f,0.0f,1.0f,1.0f,
    1.0f,0.0f,1.0f,1.0f,
    1.0f,0.0f,1.0f,1.0f,
    1.0f,0.0f,1.0f,1.0f,
    //Left White
    1.0f,1.0f,1.0f,1.0f,
    1.0f,1.0f,1.0f,1.0f,
    1.0f,1.0f,1.0f,1.0f,
    1.0f,1.0f,1.0f,1.0f,
    //Top Black
    0.0f,0.0f,0.0f,1.0f,
    0.0f,0.0f,0.0f,1.0f,
    0.0f,0.0f,0.0f,1.0f,
    0.0f,0.0f,0.0f,1.0f,
    //Bottom Yellow
    1.0f,1.0f,0.0f,1.0f,
    1.0f,1.0f,0.0f,1.0f,
    1.0f,1.0f,0.0f,1.0f,
    1.0f,1.0f,0.0f,1.0f

};
```

Ilustración 3:Captura 1

Ilustración 4:Captura 2