eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Ingeniería Informática
UPV/EHU

Documentación de Gráficos por Computador.

Mikel Molina

Gráficos por Computador
23 de septiembre de 2023

Índice

1 Objetivos de la aplicación.

El objetivo de la primera entrega de la aplicación es simple: Debe dibujar triángulos con una textura que se le pasa al programa. Las dimensiones y puntos del triángulo las pasará el usuario a través de un fichero. Sin embargo, el fichero debe seguir un formato concreto: cada línea contendrá un triángulo, y antes de escribir los puntos, se debe poner una "t" por delante. Así pues, cada punto del triángulo se escribirá de la siguiente forma:

$$t\ x_1\ y_1\ z_1\ u_1\ v_1\ x_2\ y_2\ z_2\ u_2\ v_2\ x_3\ y_3\ z_3\ u_3\ v_3$$

Donde $x_i\ y_i\ z_i\ u_i\ v_i$ son los puntos correspondientes al vértice i-ésimo del triángulo. La aplicación tenía que encargarse de dibujar el interior de los triángulos, para ello hemos hecho uso de rectas. Más adelante se explicará el método usado para lograrlo.

2 Interacción con el/la usuario/a

Las teclas disponibles son de momento limitadas, tan solo se han implementado tres teclas:

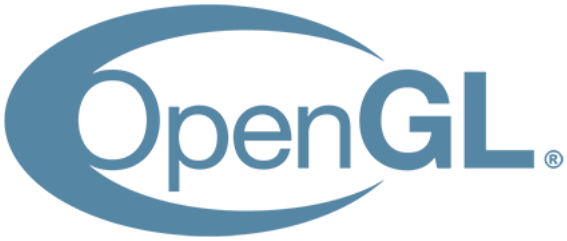
1. La tecla enter sirve para que el programa muestre el siguiente triángulo en el fichero entregado.

2. La tecla "l" sirve para cambiar a un triángulo que no enseña texturas, éste ya estaba antes de empezar con el proyecto.

3. La tecla "f" sirve para cambiar el fichero de entrada al que deseemos nosotros.

3 Herramientas utilizadas.

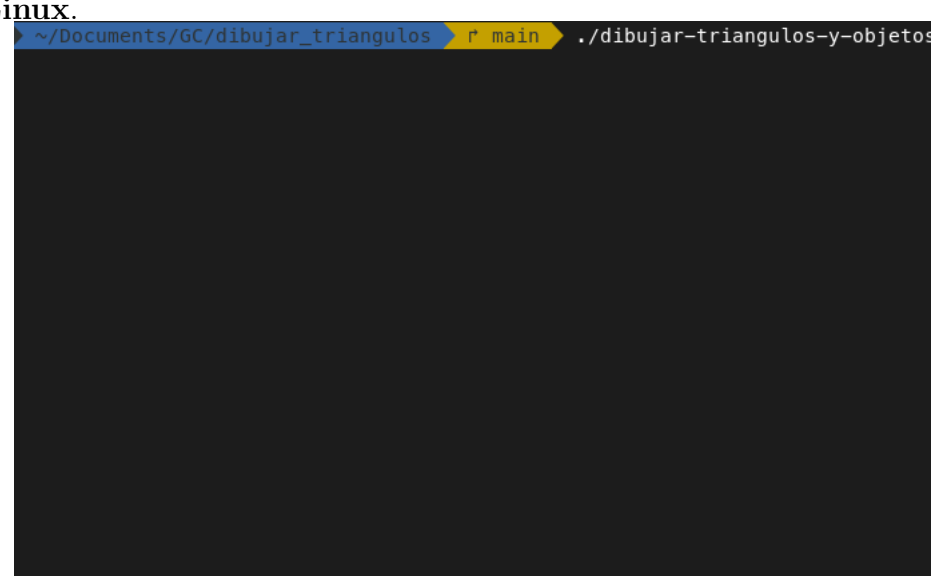
Para llevar a cabo la primera entrega del proyecto, hemos hecho uso de GLUT (OpenGL Utility Toolkit), una librería que contiene utilidades de OpenGL, pero más manejable para un programador que se esté iniciando. Perfecto para hacer este tipo de proyectos, en los que la tarea es relativamente simple. Como esta librería se encuentra en C, el programa está escrito en este lenguaje.



4 Uso de la aplicación

El programa se puede ejecutar desde una interfaz gráfica, pero es recomendado ejecutarlo desde el terminal de linux. Esto es porque la aplicación, una vez ejecutada, leerá de un fichero predeterminado, pero tenemos la capacidad de cambiarlo, y ofrecerle uno que nosotros queramos. Éste es un ejemplo de cómo se debe usar la aplicación correctamente:

1. Llamamos la aplicación desde un terminal de cualquier distribución de **Linux**.



2. Si deseamos cambiar el fichero predeterminado, pulsamos la tecla "f" y escogemos el fichero que queramos.

```
mikel@xqw ~/Documents/GC/dibujar_triangulos ᚠ main ./dibujar-triangulos-y-objetos
 Triangeluak: barneko puntuak eta testura
 Triángulos con puntos internos y textura
Press <ESC> to finish
dimentsioak irakurrita: 128,128
kolorearen zenbaki maximoa irakurrita: 255
bufferra ondo irakurri du
datuak irakurrita
Lecura finalizada
idatzi fitxategi izena
el_fichero_que_queramos.txt
```

3. El fichero debe seguir estrictamente este formato:

```
1 t -100.000000 -230.000000 0.000000 0.000000 0.000000 -81.000000 200.000000 59.000000 0.100000 1.000000 -100.000000 200.000000 0.000000 0.000000 1.000000
2 t -100.000000 -230.000000 0.000000 0.000000 0.000000 -81.000000 -230.000000 59.000000 0.100000 0.000000 -81.000000 200.000000 59.000000 0.100000 1.000000
3 t -81.000000 -230.000000 59.000000 0.100000 0.000000 -31.000000 200.000000 95.000000 0.200000 1.000000 -81.000000 200.000000 59.000000 0.100000 1.000000
4 t -81.000000 -230.000000 59.000000 0.100000 0.000000 -31.000000 -230.000000 95.000000 0.200000 0.000000 -31.000000 200.000000 95.000000 0.200000 1.000000
5 t -31.000000 -230.000000 95.000000 0.200000 0.000000 30.000000 200.000000 95.000000 0.300000 1.000000 -31.000000 200.000000 95.000000 0.200000 1.000000
6 t -31.000000 -230.000000 95.000000 0.200000 0.000000 30.000000 -230.000000 95.000000 0.300000 0.000000 30.000000 200.000000 95.000000 0.300000 1.000000
7 t 30.000000 -230.000000 95.000000 0.300000 0.000000 81.000000 200.000000 59.000000 0.400000 1.000000 30.000000 200.000000 95.000000 0.300000 1.000000
8 t 30.000000 -230.000000 95.000000 0.300000 0.000000 81.000000 -230.000000 59.000000 0.400000 0.000000 81.000000 200.000000 59.000000 0.400000 1.000000
9 t 81.000000 -230.000000 59.000000 0.400000 0.000000 100.000000 200.000000 0.000000 0.500000 1.000000 81.000000 200.000000 59.000000 0.400000 1.000000
10 t 81.000000 -230.000000 59.000000 0.400000 0.000000 100.000000 -230.000000 0.000000 0.500000 0.000000 100.000000 200.000000 0.000000 0.500000 1.000000
11 t 100.000000 -230.000000 0.000000 0.500000 0.000000 81.000000 200.000000 -59.000000 0.600000 1.000000 100.000000 200.000000 0.000000 0.500000 1.000000
12 t 100.000000 -230.000000 0.000000 0.500000 0.000000 81.000000 -230.000000 -59.000000 0.600000 0.000000 81.000000 200.000000 -59.000000 0.600000 1.000000
13 t 81.000000 -230.000000 -59.000000 0.600000 0.000000 30.000000 200.000000 -95.000000 0.700000 1.000000 81.000000 200.000000 -59.000000 0.600000 1.000000
14 t 81.000000 -230.000000 -59.000000 0.600000 0.000000 30.000000 -230.000000 -95.000000 0.700000 0.000000 30.000000 200.000000 -95.000000 0.700000 1.000000
15 t 30.000000 -230.000000 -95.000000 0.700000 0.000000 -31.000000 200.000000 -95.000000 0.800000 1.000000 30.000000 200.000000 -95.000000 0.700000 1.000000
16 t 30.000000 -230.000000 -95.000000 0.700000 0.000000 -31.000000 -230.000000 -95.000000 0.800000 0.000000 -31.000000 200.000000 -95.000000 0.800000 1.000000
17 t -31.000000 -230.000000 -95.000000 0.800000 0.000000 -81.000000 200.000000 -59.000000 0.900000 1.000000 -31.000000 200.000000 -95.000000 0.800000 1.000000
18 t -31.000000 -230.000000 -95.000000 0.800000 0.000000 -81.000000 -230.000000 -59.000000 0.900000 0.000000 -81.000000 200.000000 -59.000000 0.900000 1.000000
19 t -81.000000 -230.000000 -59.000000 0.900000 0.000000 -100.000000 200.000000 0.000000 1.000000 1.000000 -81.000000 200.000000 -59.000000 0.900000 1.000000
20 t -81.000000 -230.000000 -59.000000 0.900000 0.000000 -100.000000 -230.000000 0.000000 1.000000 0.000000 -100.000000 200.000000 0.000000 1.000000 1.000000
```

Figure 1: Nótese que cada línea contiene un triángulo, tal y como hemos dicho en la introducción.

Por último, si deseamos cambiar la imagen del programa, hay que tener en cuenta que para ello deberemos cambiar el formato de la imagen a ppm (Portabol Pixmab). Éste formato representa los valores RGB de cada píxel mediante números que van del 0 al 255.

5 Código C

A continuación tenemos el código implementeado para que el programa funcionase. Como veremos, se ha dividido en varias funciones para facilitar el entendimiento, con la falta de eficiencia que ello implica. Ésta es la función principal que debíamos cambiar.

```
void dibujar_triangulo(triobj *optr, int i)
{
  hiruki *tptr;
  punto *pgoiptr, *pbeheptr, *perdiptr;
  punto *pgoiptr2, *pbeheptr2, *perdiptr2;
  punto corte1, corte2;
  int start1, star2;
  float t = 1, s = 1, q = 1;
  float decremento_t = 1, decremento_s = 1, decremento_q = 1;
  float c1x, c1z, c1u, c1v, c2x, c2z, c2u, c2v;
  int linea;
  float cambio1, cambio1z, cambio1u, cambio1v, cambio2, cambio2z,
```

```
    cambio2u, cambio2v;
    punto p1, p2, p3;

    if (i >= optr->num_triangles)
    return;
    tptr = optr->triptr + i;
    mxp(&p1, optr->mptr->m, tptr->p1);
    mxp(&p2, optr->mptr->m, tptr->p2);
    mxp(&p3, optr->mptr->m, tptr->p3);
    if (lineak == 1)
    {
      glBegin(GL_POLYGON);
      glVertex3d(p1.x, p1.y, p1.z);
      glVertex3d(p2.x, p2.y, p2.z);
      glVertex3d(p3.x, p3.y, p3.z);
      glEnd();
      return;
    }

    //Encontramos el punto máximo, mínimo, y medio del triángulo.

    encontrar_max_min(tptr, &pgoiptr, &perdiptr, &pbeheptr);

    // Llamada a función rellenar triángulo.

    rellenar_triangulo(pgoiptr, perdiptr, pbeheptr);

  }
```

Como podemos ver, solo hemos puesto dos funciones, veamos de cerca ambas:

```
  void encontrar_max_min(hiruki *tptr, punto **pgoiptr, punto **perdiptr,
  punto **pbeheptr)
  {
    if (tptr->p1.y > tptr->p2.y)
    {
      if (tptr->p1.y > tptr->p3.y)
      {
        *pgoiptr = &(tptr->p1);
        if (tptr->p2.y > tptr->p3.y)
        {
          *perdiptr = &(tptr->p2);
          *pbeheptr = &(tptr->p3);
        }
        else
        {
          *perdiptr = &(tptr->p3);
          *pbeheptr = &(tptr->p2);
        }
      }
      else
      {
        *pgoiptr = &(tptr->p3);
        *perdiptr = &(tptr->p1);
        *pbeheptr = &(tptr->p2);
      }
    }
    else if (tptr->p2.y > tptr->p3.y)
    {
      *pgoiptr = &(tptr->p2);
      *pbeheptr = &(tptr->p1);
      *perdiptr = &(tptr->p3);
```

```
    }
    else
    {
      *pgoiptr = &(tptr->p3);
      *pbeheptr = &(tptr->p1);
      *perdiptr = &(tptr->p2);
    }
  }
```

encontrar max min busca, dado un triplete de puntos, el punto con mayor valor de y, el número con menor de y, y el número intermedio de y.