Tema 1. Introducción a la informática gráfica

Elena Cantero Molina (Grupo A)

22 de Octubre de 2018

28. ¿Podría afirmar desde el punto de vista topológico que el cubo de la Ilustración 39 se parece mucho más a una esfera que un donut?
Sí, se puede afirmar.

29.De las dos aproximaciones mostradas (glBegin/glEnd y glDrawArrays), ¿cuál cree que es más eficiente en términos de tiempo de procesamiento? ¿y en cuanto a transferencia CPU-bus-GPU?
Creo que glBegin/glEnd es más eficiente en tiempo de procesamiento, mientras que glDrawArrays es más eficiente en cuanto a transferencia CPU-bus-GPU.

31. ¿Cuántas llamadas a glVertex se realizan para una tira de n triángulos?
Si se usa glBegin/glEnd con la primitiva GL_TRIANGLE_STRIP, se hacen n llamadas a glVertex.

32. Defina una estructura de datos que permita almacenar tiras de triángulos de una malla 3D.

```
struct vector{
  int num;          //numero de triangulos totales
  Tupla3n *triangulos;
}
```

33. Escriba un código en C que calcule el área total de una malla de triángulos almacenada como lista de triángulos y vértices.

```
double AreaTotal(Malla *mesh){
    areatotal = 0;

    for( Natural i = 0; i < mesh->num_tri; i++){
        dBase = sqrt((mesh->triangulos[i][1][X]-
        mesh->triangulos[i][0][X]) + (mesh->triangulos[i][1][Y]
```

```
            - mesh->triangulos[i][0][Y]) + (mesh->triangulos[i][1][Z]
            - mesh->triangulos[i][0][Z]));

            m_base = dBase/2;

            hipo = dBase = sqrt((mesh->triangulos[i][2][X]-
            mesh->triangulos[i][1][X]) + (mesh->triangulos[i][2][Y]
            - mesh->triangulos[i][1][Y]) + (mesh->triangulos[i][2][Z]
            - mesh->triangulos[i][1][Z]));

            altura = sqrt(hipo*hipo - m_base+m_base);

            area = (dBase*altura)/2;

            areatotal += area;
        }
}
```

34. Defina la estructura de datos en C que almacene la estructura de aristas aladas. Escriba una función en C que la rellene a partir de una lista de triángulos y vértices.

```
typedef unsigned int Index;

struct Point3D{ float x, y, z; }

struct Vertice{
    Point3D p;
    Index arista;
}

struct Cara{ Index arista; }

struct Arista{
    Index V1, V2, caraIzda, caraDcha, aristaSigDcha;
    Index aristaSigIzda, aristaAntDcha, aristaAntIzda;
}

class MallaAA{
    std::vector<Vertice> v;
    std::vector<Cara> c;
    std::vector<Arista> a;

    void Relleno(Malla *mesh){
```

```
    }
}
```

35. ¿Se podría evitar tener el campo semiarista opuesta? En caso afirmativo, ¿cómo? Si no, ¿por qué?
Sí se podría evitar tener el campo semiarista opuesta. En el vector, si estoy en una semiarista par, la semiarista opuesta será la siguiente, pero si estoy en una impar, la semiarista opuesta es la anterior.

36. Defina la estructura de datos en C que almacene la estructura de semiaristas aladas.

```
typedef unsigned int Index;

struct Point3D{ float x, y, z; }

struct Vertice{
    Point3D p;
    Index semiarista;
}

struct Cara{ Index semiarista; }

struct Semiarista{
    Index V1, V2, cara, semiaristaSig, semiaristaAnt, opuesta;
}

class MallaSAA{
    std::vector<Vertice> v;
    std::vector<Cara> c;
    std::vector<Semiarista> a;
}
```

38. Considera una malla como la de la figura, en la cual hay n columnas y m filas de (pares de) triángulos
- Expresa el número de vértices en función de n y m
- Suponiendo que Natural y Real ocupan 4bytes, calcular el espacio en disco utilizando:
1. Lista de triángulos.
2. Lista de triángulos y vértices.
3. Aristas aladas.
4. Semiaristas aladas.
- vértices = (n+1)*(m+1)
1. 12*(número de triángulos) bytes
2. 12*(número de triángulos) + 12*(número de vértices) bytes

3. 56 bytes
4. 32 bytes

41. ¿Tiene influencia en el valor de la normal en el vértice el área de los triángulos que lo comparten?
Sí.

44. Calcule los nuevos valores de la posición en el espacio del punto [3,0,0] aplicándole las rotaciones del ejercicio anterior, por separado, y además tras las siguientes secuencias:

$$1.R_x[90] \rightarrow R_y[-90] \rightarrow R_z[200]$$

$$2.R_z[200] \rightarrow R_y[-90] \rightarrow R_x[90]$$

Para estas secuencias, calcule tras cada rotación el valor temporal del punto y su posición final

1ª parte:

$$[p'_x p'_y p'_z] = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(90) & \sin(90) \\ 0 & -\sin(90) & \cos(90) \end{bmatrix} = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix}$$

$$[p'_x p'_y p'_z] = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos(-90) & 0 & -\sin(-90) \\ 0 & 1 & 0 \\ \sin(-90) & 0 & \cos(-90) \end{bmatrix} = \begin{bmatrix} 0 & 0 & -3 \end{bmatrix}$$

$$[p'_x p'_y p'_z] = \begin{bmatrix} 0 & 0 & -3 \end{bmatrix} \begin{bmatrix} \cos(200) & \sin(200) & 0 \\ -\sin(200) & \cos(200) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -3 \end{bmatrix}$$

2ªparte:

$$[p'_x p'_y p'_z] = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos(200) & \sin(200) & 0 \\ -\sin(200) & \cos(200) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -0{,}9 & -0{,}34 & 0 \end{bmatrix}$$

$$[p'_x p'_y p'_z] = \begin{bmatrix} -0{,}9 & -0{,}34 & 0 \end{bmatrix} \begin{bmatrix} \cos(-90) & 0 & -\sin(-90) \\ 0 & 1 & 0 \\ \sin(-90) & 0 & \cos(-90) \end{bmatrix} = \begin{bmatrix} 0 & -0{,}34 & 0 \end{bmatrix}$$

$$[p'_x p'_y p'_z] = \begin{bmatrix} 0 & -0{,}34 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(90) & \sin(90) \\ 0 & -\sin(90) & \cos(90) \end{bmatrix} = \begin{bmatrix} 0 & 0 & -0{,}34 \end{bmatrix}$$

45. Para comprobar la no conmutatividad de las transformaciones geométricas, calcule los nuevos valores de la posición en el espacio del punto p= [3,0,0] aplicándole las siguientes secuencias:
- P'=(T3 •T2 •T1)p
- P'=(T1 •T2 •T3)p
- P'=(T3 •T1 •T2)p
Siendo
- T 1 = Rx [90]
- T 2 = T[-9,4,3]
- T 3 = Rz [45]
Para estas secuencias, calcule tras cada transformación de la composición el valor temporal del punto y su posición final. - P'=(T3 •T2 •T1)p
1º- T1

$$[p'_x p'_y p'_z] = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(90) & \sin(90) \\ 0 & -\sin(90) & \cos(90) \end{bmatrix} = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix}$$

2º- T2

$$[p'_x p'_y p'_z] = [3-9, 0+4, 0+3] = \begin{bmatrix} -6 & 4 & 3 \end{bmatrix}$$

3º - T3

$$[p'_x p'_y p'_z] = \begin{bmatrix} -6 & 4 & 3 \end{bmatrix} \begin{bmatrix} \cos(45) & \sin(45) & 0 \\ -\sin(45) & \cos(45) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1{,}41 & -1{,}41 & 3 \end{bmatrix}$$

- P'=(T1 •T2 •T3)p
1º - T3

$$[p'_x p'_y p'_z] = \begin{bmatrix} 3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos(45) & \sin(45) & 0 \\ -\sin(45) & \cos(45) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2{,}12 & 2{,}12 & 0 \end{bmatrix}$$

2º- T2

$$[p'_x p'_y p'_z] = [2{,}12-9, 2{,}12+4, 0+3] = \begin{bmatrix} -6{,}88 & 6{,}12 & 3 \end{bmatrix}$$

3º- T1

$$[p'_x p'_y p'_z] = \begin{bmatrix} -6{,}88 & 6{,}12 & 3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(90) & \sin(90) \\ 0 & -\sin(90) & \cos(90) \end{bmatrix} = \begin{bmatrix} -6{,}88 & 3 & 6{,}12 \end{bmatrix}$$

- P'=(T3 •T1 •T2)p

1º- T2

$$[p'_x p'_y p'_z] = [3-9, 0+4, 0+3] = \begin{bmatrix} -6 & 4 & 3 \end{bmatrix}$$

2º- T1

$$[p'_x p'_y p'_z] = \begin{bmatrix} -6 & 4 & 3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(90) & \sin(90) \\ 0 & -\sin(90) & \cos(90) \end{bmatrix} = \begin{bmatrix} -6 & -3 & 4 \end{bmatrix}$$

3º - T3

$$[p'_x p'_y p'_z] = \begin{bmatrix} -6 & -3 & 4 \end{bmatrix} \begin{bmatrix} \cos(45) & \sin(45) & 0 \\ -\sin(45) & \cos(45) & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -2{,}12 & -6{,}36 & 4 \end{bmatrix}$$

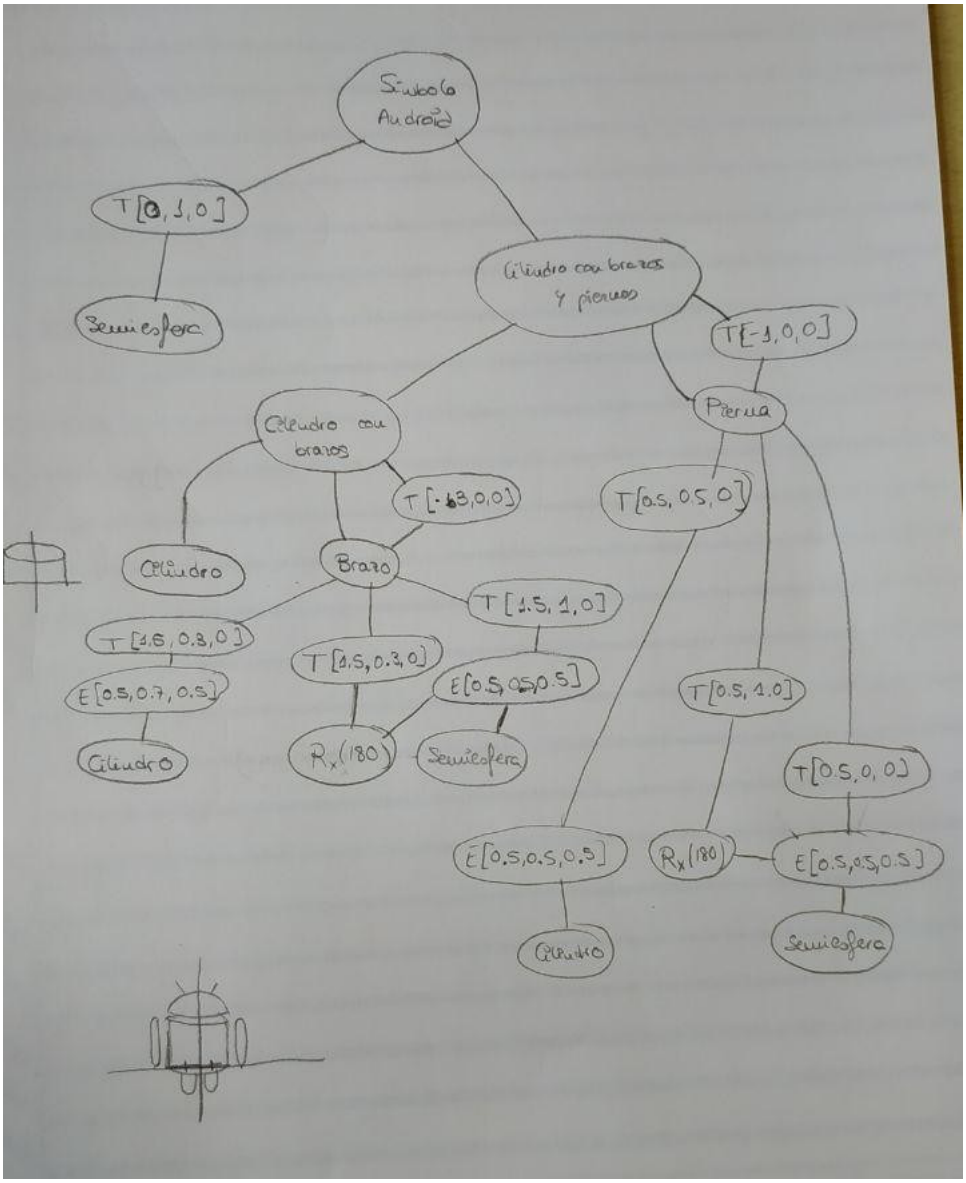
49. Documente en sus apuntes: ¿Cómo se hace una rotación con respecto a cualquier punto?

Primero hay que trasladar el punto al origen de coordenadas, después se rota lo que se desee y por último se traslada de nuevo al lugar de origen.

51. Supón que dispones de dos objetos simples: Semiesfera y Cilindro, con su correspondiente función dibujar(). La semiesfera (en coordenadas maestras) tiene radio unidad, centro en el origen y el eje vertical en el eje Y. Igualmente el cilindro tiene radio y altura unidad, el centro de la base está en el origen, y su eje es el eje Y. Con estas dos primitivas queremos construir la figura símbolo de Android:

- Diseña el grafo de escena correspondiente, ten en cuenta que hay objetos compuestos que se pueden instanciar más de una vez (cada brazo o pierna se puede construir con un objeto compuesto de dos semiesferas en los extremos de un cilindro).

- Escribe el código OpenGL para visualizarlo, usando transformaciones y push/pop de la matriz modelview.




```
Android::dibuja(){
    glPushMatrix();
        glPushMatrix();
            glPushMatrix();
                Cilindro.dibuja();
                glTranslatef(-3.0, 0.0, 0.0);
                glPushMatrix();
                    glTanslatef(1.5, 0.3, 0.0);
                    glScalef(0.5, 0.7, 0.5);
                    Cilindro.dibuja();
```

```
                    glTranslate(1.5, 0.3, 0.0);
                    glRotatef(180);
                    glScalef(0.5, 0.5, 0.5);
                  Semiesfera.dibuja();
                  glTranslatef(1.5, 1.0, 0.0);
                  glScalef(0.5, 0.5, 0.5);
                  Semiesfera.dibuja();
               glPopMatrix();
            glPopMatrix();
            glTranslatef(-1.0, 0.0, 0.0);
            glPushMatrix();
                glTranslatef(0.5, 0.5, 0.0);
                glScalef(0.5, 0.5, 0.5);
                Cilindro.dibuja();
                glTranslatef(0.5, 1.0, 0.0);
                glRotatef(180);
                glScalef(0.5, 0.5, 0.5);
                Semiesfera.dibuja();
                glTranslatef(0.5, 0.0, 0.0);
                glScalef(0.5, 0.5, 0.5);
                Semiesfera.dibuja();
            glPopMatrix();
        glPopMatrix();
        glTanslatef(0.0, 1.0, 0.0);
        Semiesfera.dibuja();
    glPopMatrix();
}
```