UNIVERSIDAD DE CÓRDOBA
ESCUELA POLITÉCNICA SUPERIOR
DEPARTAMENTO DE INFORMÁTICA Y ANÁLISIS NUMÉRICO

4º Grado en ingeniería informática
Informática Gráfica

Modelo articulado con OpenGL

Francisco García Díaz
Jesús Rodríguez Pérez

INDICE DE CONTENIDOS

a) Algoritmo de generación del esqueleto

El esqueleto se genera a través de la función init la cual crea cada join.

```
Vector3f pos(0,0,0);
Vector3f toplimit(0,0,0);
Vector3f botlimit(0,0,0);
Join j(0,0,pos,toplimit,botlimit);
joins.push_back(j);
```

Ejemplo de creación y adición al esqueleto de un join.

Se crean 3 vectores para inicializar la posición y los limites de giro que tendrá el join y luego este join se introduce en un array de joins que son los joins que conforman nuestro esqueleto.

Para generar el esqueleto se llama a la función de dibujado dibujaEscena que se encarga de dibujar cada join. Para hacer estos dibujos se han usado dos funciones auxiliares, las que nos permiten dibujar un punto y una linea:

```
void punto(int j){
	glBegin(GL_POINTS);
		glVertex3f(joins[j].pos.x, joins[j].pos.y, joins[j].pos.z);
	glEnd();
}

void linea(int j){
	glBegin(GL_LINES);
		glVertex3f(joins[j].pos.x, joins[j].pos.y, joins[j].pos.z);
		int r=joins[j].root;
		glVertex3f(joins[r].pos.x, joins[r].pos.y, joins[r].pos.z);
	glEnd();
}
```

Después es tan simple como llamar a cada uno de nuestros join en la función de dibujar escena haciendo que dibuje su punto y que dibuje una linea desde si mismo hasta el nodo padre.

Los giros de cada join se indicarán en esta función.

```
void rotate(Vector3f rot){
	Vector3f rotacion;
	rotacion.devuelveMenor(topLimit,rot);
	rotacion.devuelveMayor(botLimit,rotacion);
	angle=rotacion;
}
```

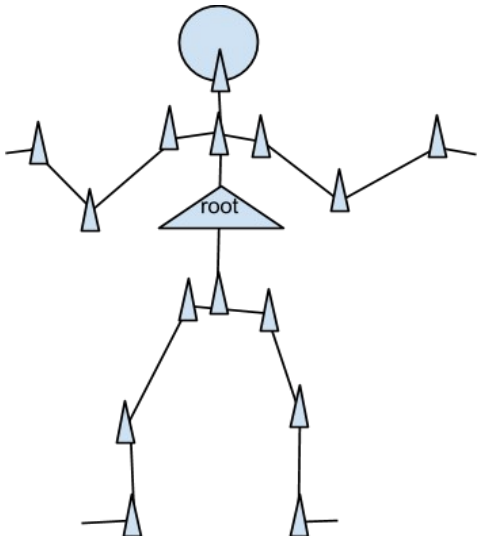
b) Estructura de datos. Jerarquia de bones y joins

Los join son una clase creada que sigue la siguiente estructura:
Variables:

```
int id              // Identificador del join
int root                    // Identificador del padre
Vector3f toplimit   // Limite superior
Vector3f botlimit   // Limite inferior
Vector3f angle              // Angulo de giro actual
Vector3f pos        // Posición del join
```

Vector3f es una clase que contiene 3 variables float.

La jerarquia de joins es la siguiente:



Cada join tendrá un identificador único que permitirá a través de una función llamarlo para ser girado.
Los identificadores están declarados como defines en el código.

c) Tabla de grados de libertad (DoF)

Nodo	Rx	Ry	Rz
0 - ROOT*	-360 – 360	-360 – 360	-360 – 360
1 - TORSO	0 – 0	-90 – 90	-40 – 40
2 - CUELLO	- 60 – 20	-0 – 0	-30 – 30
3 - CINTURA	-30 – 30	20 – 20	-10 – 10
4 - CABEZA	-360 – 360	-360 – 360	-360 – 360
5 - HOMBRO_DER	-90 – 150	-60 – 120	-90 – 80
6 - HOMBRO_IZQ	-150 – 90	-120 – 60	-90 – 80

7 - CADERA_DER	-30 – 50	-10 – 10	0 – 0
8 - CADERA_IZQ	-30 – 50	-10 – 10	0 – 0
9 - CODO_DER	0 – 0	0 – 0	0 – 140
10 - CODO_IZQ	0 – 0	0 – 0	0 – 140
11 - MUNECA_DER	-360 – 360	-30 – 30	-50 – 50
12 - MUNECA_IZQ	-360 – 360	-30 – 30	-50 – 50
13 - MANO_DER	-360 – 360	-360 – 360	-360 – 360
14 - MANO_IZQ	-360 – 360	-360 – 360	-360 – 360
15 - RODILLA_DER	-60 – 0	0 – 0	0 – 0
16 - RODILLA_IZQ	-60 – 0	0 – 0	0 – 0
17 - TALON_DER	0 – 0	0 – 0	0 – 0
18 - TALON_IZQ	0 – 0	0 – 0	0 – 0
19 - PIE_DER	-360 – 360	-360 – 360	-360 – 360
20 - PIE_IZQ	-360 – 360	-360 – 360	-360 – 360

*El nodo ROOT además de los tres grados de libertad de rotación tiene tres mas de translación.

d) Movimiento del esqueleto. Primitivas

- Rotación de un join:
  Se encarga de asignar un ángulo a un join del esqueleto.

```
void rotate(Vector3f rot);
```

Ejemplo de uso:

```
e.joins[TORSO].rotate(Vector3f(50.0f,0,0));
```

Se rota el join del roso 50 grados en el eje X

- Translación del esqueleto:
  Permite mover la posición global del esqueleto.

```
void move(float x, float y, float z)
```

e) Obtención y/o generación de skin.

Los puntos de la skin se obtienen a partir de la función dibujarVertices la que recorre todos los triangulos de la figura.

```
glBegin(GL_TRIANGLES);
        for(int i=0;i<asignacion[id].size();i++){
                int itri=asignacion[id][i];
                triangle = &T(group->triangles[itri]);
                glVertex3fv(&model->vertices[3 * triangle->vindices[0]]);
                glVertex3fv(&model->vertices[3 * triangle->vindices[1]]);
                glVertex3fv(&model->vertices[3 * triangle->vindices[2]]);
        }
glEnd();
glPopMatrix();
```

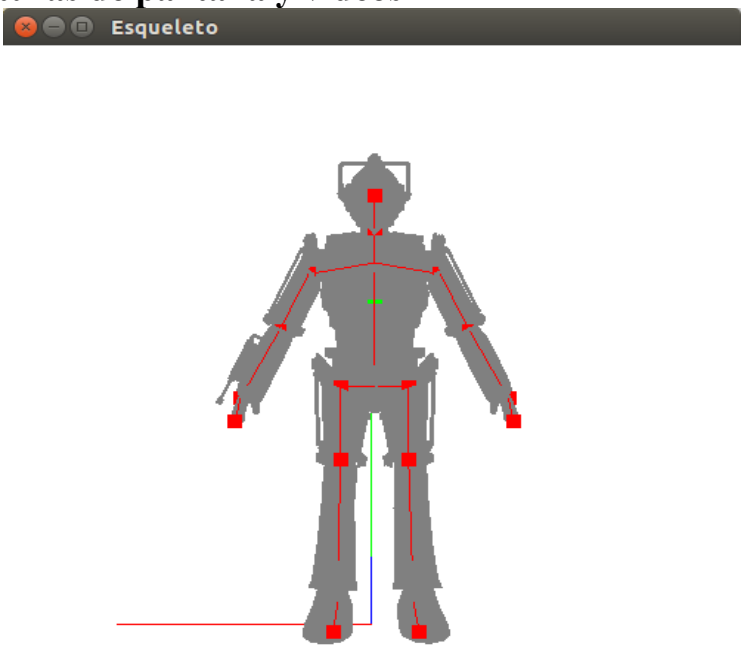
f) Algoritmo de acoplamiento de skin al esqueleto.

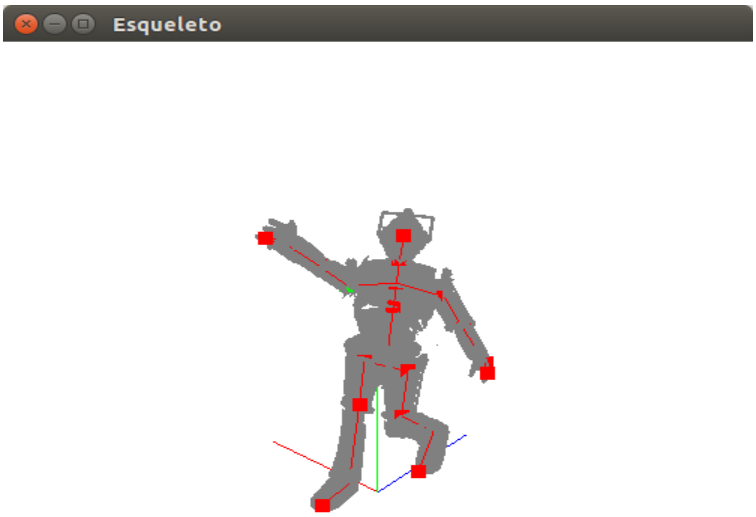
Este algoritmo es el indicado en asignarPuntos y consiste en un doble bucle que recorre todos los joins y todos los triángulos de la figura, se calculan los puntos que junto a su padre son mas próximos a estos triángulos, estos quedan asignados a ese join y al rotar ese join se ejecuta la rotación también en esos elementos.

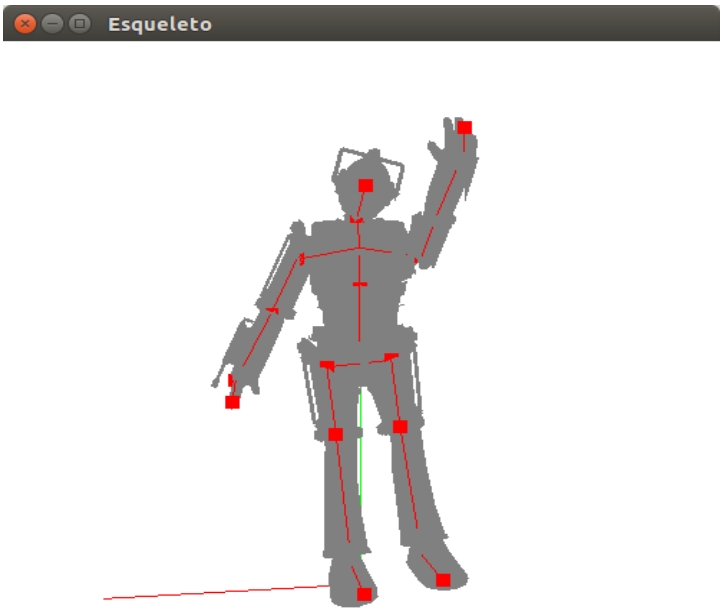
```
for (int i = 0; i < group->numtriangles; i++) {
                        triangle = &T(group->triangles[i]);
                        min=distanciaPuntoTriangulo(triangle,joins[0].pos);
                        indiceSelect=0;
                        for(int k=1;k<joins.size();k++){
                                Vector3f p((joins[k].pos.x+joins[joins[k].root].pos.x)/2.0,

        (joins[k].pos.y+joins[joins[k].root].pos.y)/2.0,

        (joins[k].pos.z+joins[joins[k].root].pos.z)/2.0
                                                                        );
                          dist=distanciaPuntoTriangulo(triangle,p);
                          if(min>dist){
                                min=dist;
                                indiceSelect=k;
```

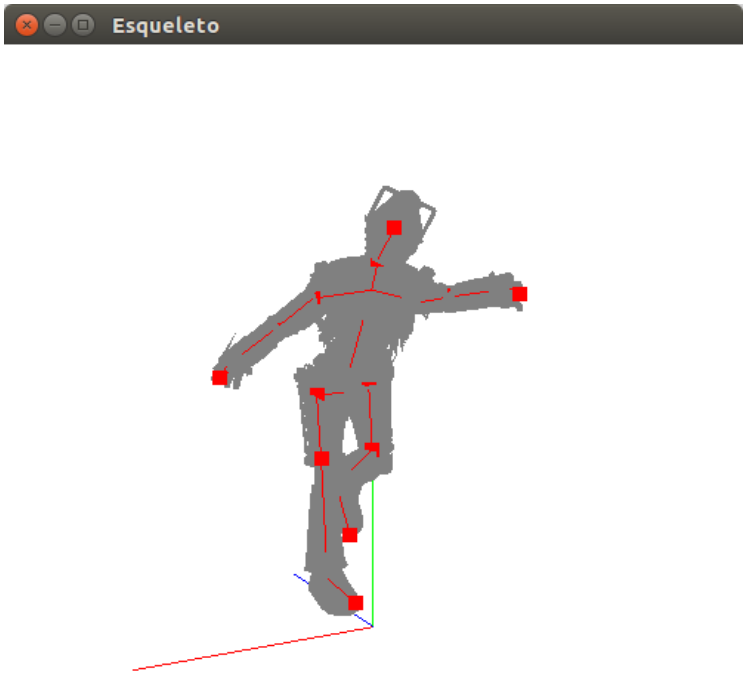
```
                    }
                }
                asignacion[indiceSelect].push_back(i);
        }
```

g) Ejemplos: capturas de pantalla y vídeos






Esqueleto


Esqueleto