Gráficos y Multimedia

Sesión 2: Proyecciones.

1. Proyección Perspectiva

OPENGL

- glFrustum(l, r, b, t, n, f); left, right, bottom, top, near, far
- gluPerspective(fovy, aspect, n, f);
- R se define si l ≠ r, b ≠ t, n ≠ f, n ≠ 0, n y f > 0

$$\mathbf{R} = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad \textbf{and } \mathbf{R}^{-1} = \begin{bmatrix} \frac{r-l}{2n} & 0 & 0 & \frac{r+l}{2n} \\ 0 & \frac{t-b}{2n} & 0 & \frac{t+b}{2n} \\ 0 & 0 & 0 & -1 \\ 0 & 0 & \frac{-(f-n)}{2fn} & \frac{f+n}{2fn} \end{bmatrix}$$

2. Proyección Paralela

OPENGL

- glOrtho(l, r, b, t, n, f); left, right, bottom, top, near, far
- R se define si l ≠ r, b ≠ t, n ≠ f, n ≠ 0, n y f > 0

$$\mathbf{R} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{-2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{R}^{-1} = \begin{bmatrix} \frac{r-l}{2} & 0 & 0 & \frac{r+l}{2} \\ 0 & \frac{t-b}{2} & 0 & \frac{t+b}{2} \\ 0 & 0 & \frac{f-n}{-2} & \frac{n+f}{2} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3. Proyecciones en OpenGL

Definición de la matriz de proyección en OpenGL

- Podemos usar proyección perspectiva:
  - void **glFrustum**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble znear, GLdouble zfar);
  - void **gluPerspective**(GLdouble fovy, GLdouble aspect, GLdouble zNear, GLdouble zFar);
- o proyección paralela:
  - void **glOrtho**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble zNear, GLdouble zFar);

4. Proyección de Vértices

Definición de la matriz de proyección en OpenGL

- glMatrixMode(GL_PROJECTION);
- glLoadIdentity();
- glFrustum(…) o glOrtho(…)

Proyección de los Vértices (orden correcto)

- $V' = M_{Projection} \cdot M_{ModelView} \cdot V$
- $V' = A \cdot V, \quad A = M_{Projection} \cdot M_{ModelView}$

3. Proyecciones en OpenGL | ES

Definición de la matriz de proyección en OpenGL|ES

- Podemos usar proyección perspectiva:
  - public static void **frustumM** (float[] m, int offset, float left, float right, float bottom, float top, float near, float far)
  - public static void **perspectiveM** (float[] m, int offset, float fovy, float aspect, float zNear, float zFar)
- o proyección paralela:
  - public static void **orthoM** (float[] m, int mOffset, float left, float right, float bottom, float top, float near, float far)

4. Proyección de Vértices

Definición de la matriz de proyección en OpenGL|ES

```
if (width > height) {
    // Landscape, TAM=tamaño deseado a visualizar
    orthoM(projectionMatrix, 0, -aspectRatio*TAM, aspectRatio*TAM, -TAM, TAM,
            -10.0f, 10.0f);
    //frustumM(projectionMatrix, 0, -aspectRatio*TAM, aspectRatio*TAM, -TAM,
        TAM, 0.1f, 100.0f);
} else {
    // Portrait or square
    orthoM(projectionMatrix, 0, -TAM, TAM, -aspectRatio*TAM, aspectRatio*TAM,
            -10.0f, 10.0f);
    //frustumM(projectionMatrix, 0, -TAM, TAM, -aspectRatio*TAM,
        aspectRatio*TAM, 0.1f, 100.0f);
}
```

4. Proyección de Vértices

Definición de la matriz de proyección en OpenGL|ES

- Debido a un bug en Android comprobamos que *frustumM()* tiene un fallo en el código para el cálculo de la primera fila, segunda columna, este valor aparece multiplicado por 2, por tanto hay que corregirlo:

```
void frustum(float[] m, int offset, float l, float r, float b, float t,
        float n, float f)
{
    frustumM(m, offset, l, r, b, t, n, f);
    // Corrección del bug de Android, orden por columnas en java
    m[8] /= 2;
}
```

En Java las matrices siguen un orden por columnas.

$$\begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

4. Proyección de Vértices

Definición de la matriz de proyección en OpenGL|ES

- Otra opción es crearnos nuestra propia función `perspective`, de la siguiente forma, siendo a la distancia focal de la cámara 1/tangente(fovy/2), debe ser menor a 180º:

```
void perspective(float[] m, int offset, float fovy, float aspect,
                 float n, float f)
{     final float d = f-n;
final float angleInRadians = (float) (fovy * Math.PI / 180.0);
final float a = (float) (1.0 / Math.tan(angleInRadians / 2.0));

m[0] = a/aspect;
m[1] = 0f;
m[2] = 0f;
m[3] = 0f;

m[4] = 0f;
m[5] = a;
m[6] = 0f;
m[7] = 0f;
```

$$\begin{bmatrix} \frac{a}{aspect} & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

4. Proyección de Vértices

Definición de la matriz de proyección en OpenGL|ES

```
        m[8] = 0;
        m[9] = 0;
        m[10] = (n - f) / d;
        m[11] = -1f;
        m[12] = 0f;
        m[13] = 0f;
        m[14] = -2*f*n/d;
        m[15] = 0f;
    }
```

Existe el método `perspectiveM()` pero sólo desde *Ice Cream Sandwitch (API 14)*, lo cual nos limitaría la distribución de nuestra app en versiones inferiores.

$$\begin{bmatrix} \frac{a}{aspect} & 0 & 0 & 0 \\ 0 & a & 0 & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

3. Proyecciones en OpenGL | ES

Definición de la matriz de proyección en OpenGL|ES

- También podemos usar el método:

```
public static void setLookAtM (float[] rm, int rmOffset, float eyeX,
        float eyeY, float eyeZ, float centerX, float centerY, float centerZ,
        float upX, float upY, float upZ)
```

Añadido en el API level 8

- Define una trasformación de vista a partir de un punto de observador, el centro de la vista y el vector vertical:

Rm:	resultado.
mOffset:	indice donde comienza la matriz.
eyeX:	coordenada X del observador.
eyeY:	coordenada Y del observador.
eyeZ:	coordenada Z del observador.
centerX:	coordenada X del centro de la vista.
centerY:	coordenada Y del centro de la vista.
centerZ:	coordenada Z del centro de la vista.
upX:	coordenada X del vector vertical.
upY:	coordenada Y del vector vertical.
upZ:	coordenada Z del vector vertical.

3. Proyecciones en OpenGL | ES

Definición de la matriz de proyección en OpenGL|ES

- Podemos encontrar la definición completa de la clase Matrix de OpenGL|ES en este enlace:
  - http://developer.android.com/reference/android/opengl/Matrix.html