Transformaciones

Resumen transformaciones con matrices

¿Qué tipo de matemáticas deben saber los desarrolladores de juegos?

La parte de matrices

Biblioteca GLMatrix

Como ayuda a la programación, la biblioteca GLMATRIX proporciona funciones tanto para la construcción de las matrices de transformación como para operar con ellas.

Accede a glMatrix. Ve a la documentación y, por ejemplo, busca el método fromTranslation de la clase mat4. La propia ayuda te permite acceder al código fuente de la librería. Ve al código del método fromTranslation y comprueba cómo construye la respectiva matriz de transformación

Biblioteca GLMatrix

Por ejemplo, la matriz de transformación M que escala un objeto al doble de su tamaño y después lo traslada en dirección del eje X un total de diez unidades, se obtendría de la siguiente forma:

```
var M = mat4.create();
var T = mat4.create();
var S = mat4.create();
mat4.fromTranslation (T, [10,0,0]);
mat4.fromScaling     (S, [2,2,2]);
mat4.multiply        (M, T, S);
```

Biblioteca GLMatrix

Por ejemplo, la matriz de transformación M que escala un objeto al doble de su tamaño y después lo traslada en dirección del eje X un total de diez unidades, se obtendría de la siguiente forma:

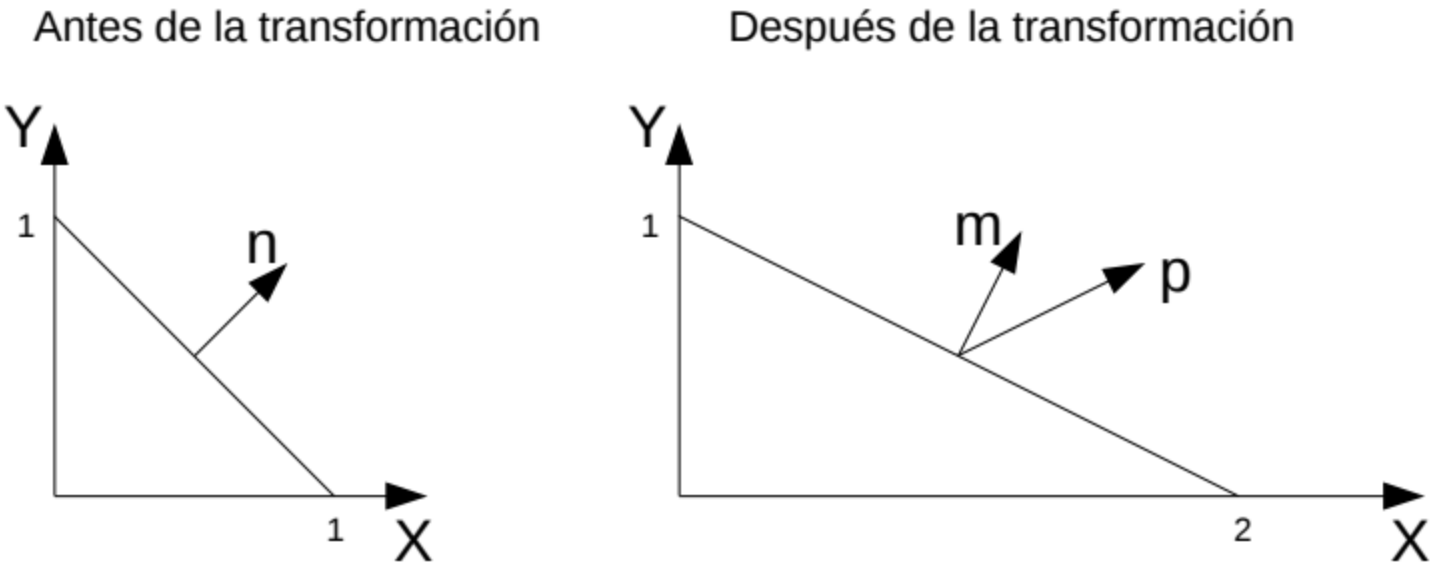
```
var M = mat4.create();
var T = mat4.create();
var S = mat4.create();
mat4.fromTranslation (T, [10,0,0]);
mat4.fromScaling     (S, [2,2,2]);
mat4.multiply        (M, T, S);
```

Biblioteca GLMatrix

¿Qué transformación produces si en el ejemplo anterior, en vez de mat4.multiply (M, T, S), apareciese mat4.multiply (M, S, T)?

Biblioteca GLMatrix

La normal



Biblioteca GLMatrix

La normal

Si M es la matriz de transformación del modelo, para obtener la matriz de transformación de la normal, N, utilizando GLMATRIX se puede hacer, por ejemplo, lo siguiente

```
var N = mat3.create();
mat3.fromMat4  (N,M);
mat3.invert    (N,N);
mat3.transpose (N,N);
```

GLMATRIX también porporciona la función normalFromMat4(N, M) que realiza exactamente las mismas operaciones que las del listado anterior.

Transformaciones en WebGL

WebGL no proporciona modelos de primitivas geométricas 3D. Por lo tanto, y en primer lugar, es necesario obtener una descripción geométrica de primitivas básicas como el cubo, la esfera, el cono, etc. De momento se utilizan modelos que solo constan de geometría, es decir, no contienen otros atributos (normal, color, etc.), por lo que vamos a visualizarlos en alambre, es decir, solo las aristas

```
function draw(model) {

  gl.bindBuffer (gl.ARRAY_BUFFER, model.idBufferVertices);
  gl.vertexAttribPointer (vertexPositionAttribute, 3,
                          gl.FLOAT, false, 0, 0);

  gl.bindBuffer (gl.ELEMENT_ARRAY_BUFFER, model.idBufferIndices);
  for (var i = 0; i < model.indices.length; i += 3)
    gl.drawElements (gl.LINE_LOOP, 3, gl.UNSIGNED_SHORT, i*2);

}
```

Transformaciones en WebGL

Drawscene() Esta función incluye ahora los tres pasos necesarios para dibujar un modelo que requiere de una matriz de transformación:

- 1. Se obtiene la matriz de transformación del modelo, modelMatrix, que en este caso se trata de un escalado a la mitad de su tamaño.
- 2. Se establece el valor de la matriz de transformación en el shader de vértices.
- 3. Se ordena el dibujado del modelo. Será entonces cuando cada vértice del modelo se multiplique por la matriz de transformación especificada en el paso anterior.

Transformaciones en WebGL

```
function drawScene() {

  gl.clear(gl.COLOR_BUFFER_BIT);

  // 1. calcula la matriz de transformación
  var modelMatrix  = mat4.create();
  mat4.fromScaling (modelMatrix, [0.5,0.5,0.5]);

  // 2. establece la matriz modelMatrix en el shader de vértices
  gl.uniformMatrix4fv(program.modelMatrixIndex, false,
      modelMatrix);

  // para la matriz de la normal:
  // var normalMatrix  = mat3.create();
  // mat3.normalFromMat4 (normalMatrix, modelMatrix);
  // gl.uniformMatrix3fv(program.normalMatrixIndex, false,
      normalMatrix);

  // 3. dibuja la primitiva
  draw(exampleCube);

}
```

Transformaciones en WebGL

En el shader de vértices se incluye la operación que multiplica cada vértice por su correspondiente matriz de transformación

```
#version 300 es

uniform mat4 M;     // matriz de transformación del modelo
// uniform mat3 N; // matriz de transformación de la normal

in vec3 VertexPosition;
// in  vec3 VertexNormal;
// out vec3 VertexNormalT;

void main () {

  // VertexNormalT = normalize (N * VertexNormal);
  gl_Position = M * vec4(VertexPosition, 1.0);

}
```

Ejercicios

1. Edita c03transforma.html y c03transforma.js. Comprueba que se han incluido todos los cambios explicados en esta sección. Con la IA (o por tu cuenta si quieres) prueba a crear diferentes primitivas como el cubo, la esfera, el cono, etc y prueba a dibujarlas.

Ejercicios

2. Se dispone del modelo de un cubo definido con su centro en el origen de coordenadas, lado 1, y sus aristas paralelas a los ejes de coordenadas. Dado el siguiente fragmento de código en el que se especifican una serie de transformaciones geométricas, dibuja una vista frontal XY (no hace falta que dibujes la Z) de cómo quedan los dos cubos que se ordena dibujar teniendo en cuenta dichas transformaciones (asume que la funcion drawCubo() produce el dibujado del modelo del cubo).

```
var S  = mat4.create();
var T1 = mat4.create();
var T2 = mat4.create();
var M  = mat4.create();

mat4.fromScaling (S, [ 1, 2, 1]);
mat4.fromTranslation (T1, [ 0, 0.5, 0]);
ma4.multiply (M, S, T1);
gl.uniformMatrix4fv(program.modelMatrixIndex, false, M);
drawCubo();

mat4.fromTranslation (T2, [ 1, 0, 0]);
ma4.multiply (M, M, T2);
gl.uniformMatrix4fv(program.modelMatrixIndex, false, M);
drawCubo();
```

Ejercicios

3. Observa la escena que se muestra en la figura 3.4. Crea una escena que produzca la misma salida utilizando únicamente la primitiva cubo. En primer lugar, piensa sobre el papel qué transformaciones necesitas aplicar y solo después procede con la escritura del código. El cubo central tiene de lado 0.1, y los que están a su lado tienen la misma base pero una altura que va incrementándose en 0.1 a medida que se alejan del central. Usa un bucle para pintar los trece cubos. Nota: ambas imágenes muestran la misma escena, pero en la imagen de la derecha se ha girado la escena para apreciar mejor que son objetos 3D.

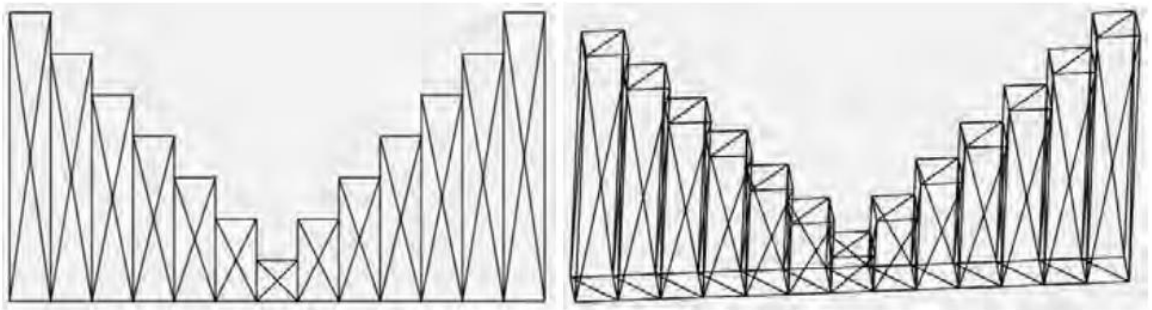
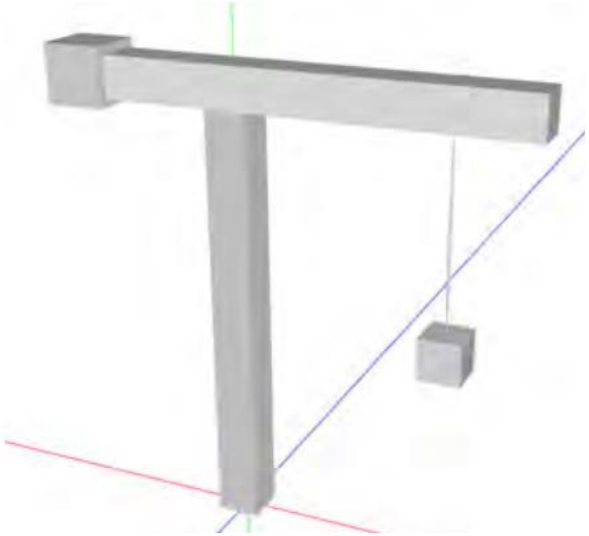


Figura 3.4: Escena modelada a partir de la primitiva geométrica cubo

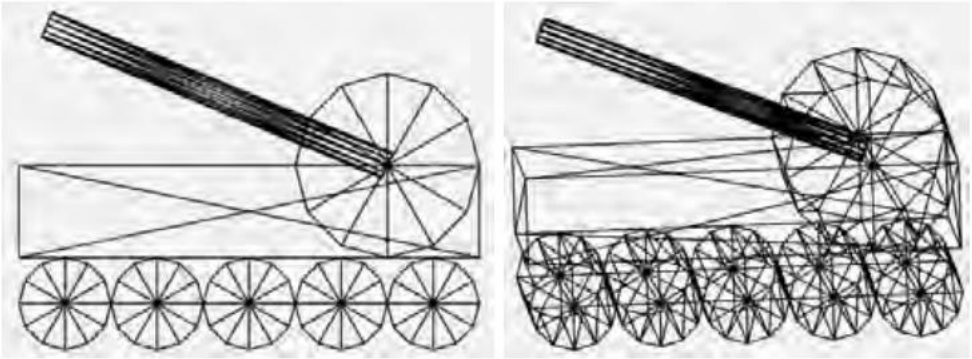
Ejercicios

4. Modela la típica grúa de obra cuyo esquema se muestra en la figura 3.5 utilizando como única primitiva cubos de lado 1 centrados en el origen de coordenadas. La carga es de lado 1, el contrapeso es de lado 1.4, y tanto el pie como el brazo tienen 10 de longitud. Incluye las transformaciones que giran la grúa sobre su pie, que desplazan la carga a lo largo del brazo, y que levantan y descienden la carga.



Ejercicios

5. En este ejercicio vas a hacer un tanque. En la figura 3.6 puedes observar lo que has de conseguir. Los cilindros de las ruedas tienen una altura de 0.8 y un radio de 0.1. Tapa los cilindros por ambos lados. El cuerpo del tanque es un cubo escalado para ocupar lo mismo que las ruedas, y una altura de 0.2. La torreta del cañón es un cilindro de radio 0.2 y altura 0.8. Observa dónde está situada y haz coincidir la tuya en la misma posición. Tápala también. Por último, añade el cañón, que es un cilindro de longitud 0.8 y radio 0.03.



Ejercicios

6. Internet es una fuente de inspiración excepcional, utilízala para buscar ejemplos que te sirvan de muestra y practicar con el uso de las transformaciones geométricas. Fíjate en los que aparecen en la figura 3.8.