Tema 6: Texturas avanzadas

José Ribelles

Departamento de Lenguajes y Sistemas Informáticos, Universitat Jaume I

SIU020 - Síntesis de Imagen y Animación

Contenido

Hoy veremos...

Texturas 3D

Descripción

- Piensa en un bloque de material de piedra, madera, etc.

- De la misma manera, una textura 3D define un valor para cada punto en el espacio 3D.

Texturas 3D

Características

- Son una extensión directa de las texturas 2D donde ahora se utilizan tres coordenadas (s, t, r).
- Ahora tenemos voxels en lugar de texels!!!
- Las propias coordenadas de los vértices se pueden utilizar como coordenadas de textura, es decir, no es necesario la parametrización de la superficie.
- Pero son caras de almacenar ... y también de filtrar (4x4 muestras).
- Ineficiente para superficies, la mayor parte de los voxels no se utilizan.
- WebGL 2.0 soporta este tipo de texturas así como los mismos tipos de filtrado, (pero WebGL 1.0 no).

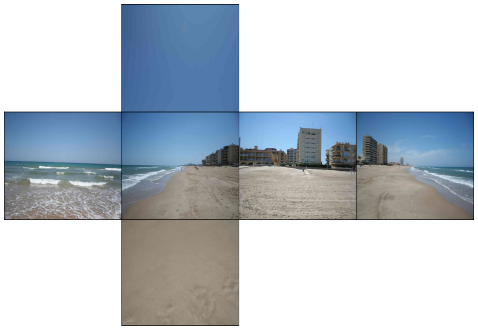
Hoy veremos...

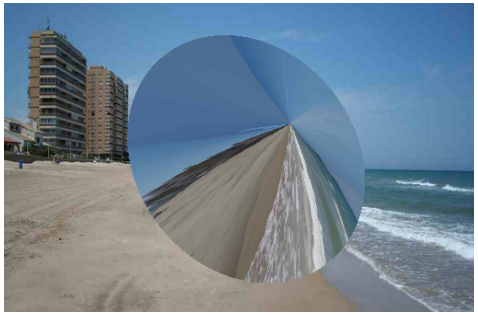
Cube Maps

Descripción

- Seis texturas cuadradas que juntas capturan un entorno.
- Cada una se pega a una cara de un cubo.
- Soportado en WebGL.
- ¿Para qué sirve?
  - Reflection maps (environment maps)
  - Refraction maps
  - Skyboxes

Reflection mapping

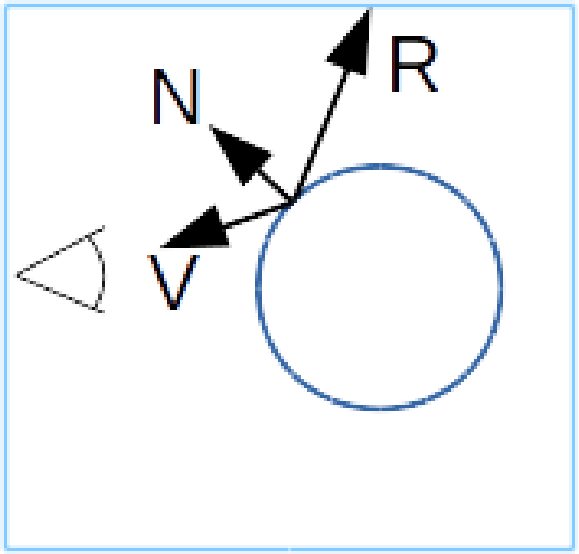




Reflection mapping

Características

- El objetivo es simular objetos que reflejan su entorno.
- Las coordenadas de textura no se suministran, se calculan!!
- Y cambian con la posición del observador.



Reflection mapping

¿Cómo se obtiene el texel?

- Para cada punto de la superficie del objeto reflejante se obtiene el vector de reflexión respecto a la normal en ese punto de la superficie.
- ¿Cuál de las seis texturas se ha de utilizar? Se elige la coordenada de mayor magnitud. Si es la coordenada R_x, se utilizan la cara derecha o izquierda del cubo, dependiendo del signo.
- Después, hay que obtener las coordenadas s y t para acceder a la textura seleccionada.
- Ejemplo en el caso de ser R_x:

$$s = (\frac{-R_z}{|R_x|} + 1)/2 \qquad t = (\frac{-R_y}{|R_x|} + 1)/2 \tag{1}$$

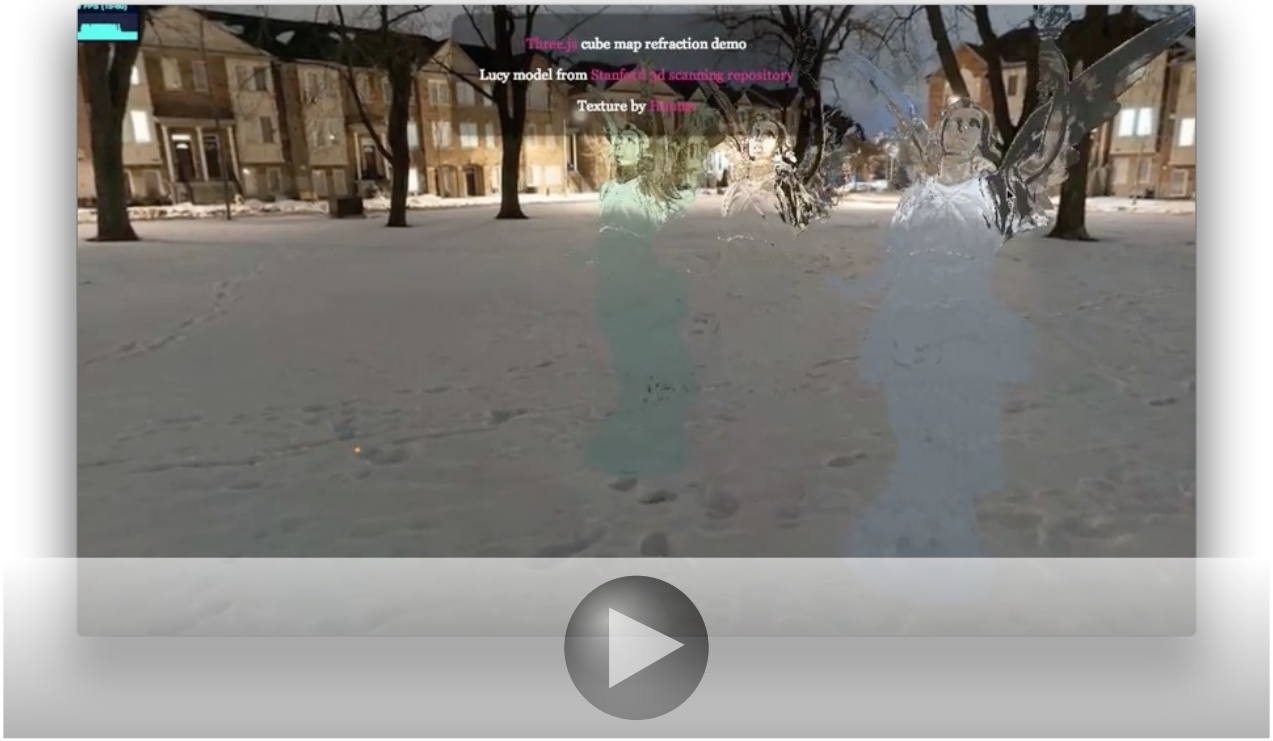
El Shader para Reflection mapping

Listado 1: Shader para Reflection Mapping

```
// Vertex shader
...
in  vec3 VertexNormal;
out vec3 R;

void main()  {
   ...
   // igual que en el modelo de iluminacion de Phong
   vec4 ecPosition = modelViewMatrix * vec4(vertexPosition,1.0);
   vec3 N          = normalize(normalMatrix * vertexNormal);
   vec3 V          = normalize(vec3(-ecPosition));

   ...
   R = reflect(-V, N);
}

// Fragment shader
...
uniform samplerCube myCubeMapTexture;     // la textura

in vec3 R;

void main()
{
   ...
   // acceso a la textura para obtener un valor de color RGBA
   fragmentColor = texture(myCubeMapTexture, R);
}
```

Video ejemplo

Refraction Mapping



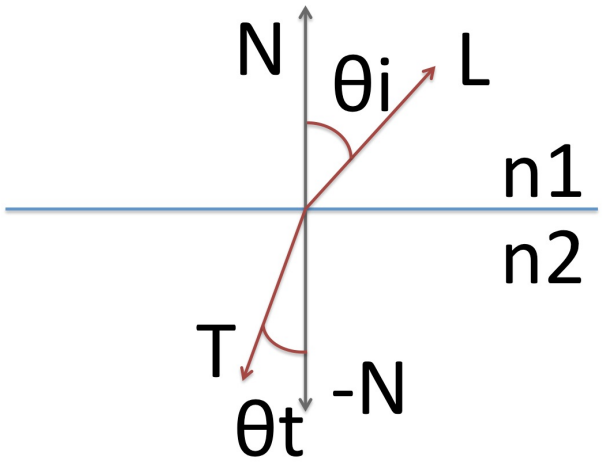
Objetivo

- Representar objetos que la luz atraviesa: hielo, agua, vidrio, esmeralda, rubí, diamente, etc.

Refraction Mapping

Cálculo

- Se utiliza la ley de Snell: $n_1 \cdot \sin\theta_i = n_2 \cdot \sin\theta_t$
- Al cociente entre n_1 y n_2 se le denomina índice de refracción.
- Por ejemplo: aire (1.0), agua (1.33), vidrio (1.52), diamante (2.42), etc.

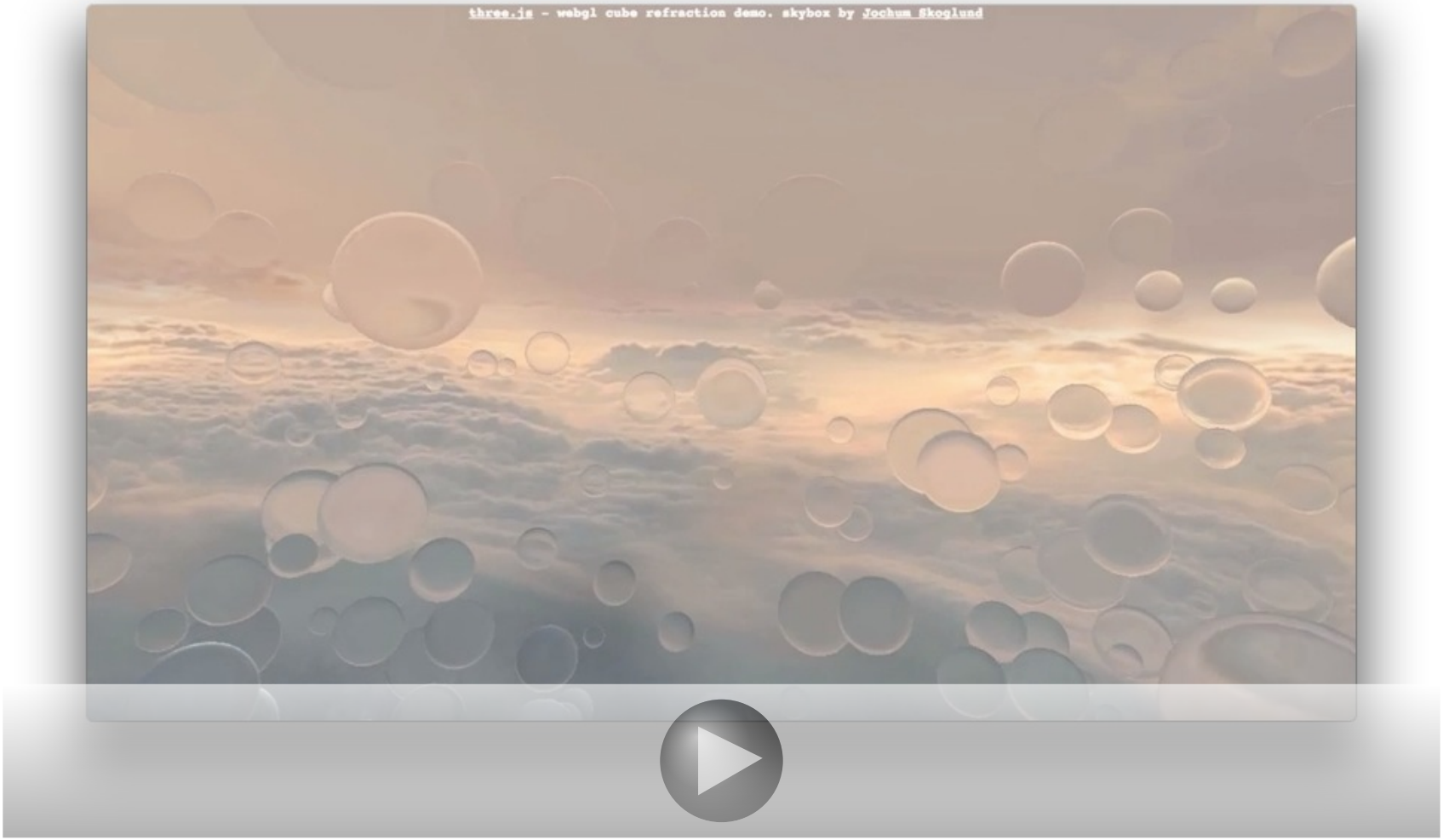


El Shader para Refraction Mapping

Listado 2: Shader para Refraction Mapping

```
// Vertex shader
...
in  vec3 VertexNormal;
out vec3 RefractDir, ReflectDir;

void main()  {
  ...
  ReflectDir = reflect(-V, N);
  RefractDir = refract(-V, N, material.indice_de_refraccion);
}

// Fragment shader
...
uniform samplerCube myCubeMapTexture;     // la textura

in vec3 RefractDir, ReflectDir;

void main()
{
  ...
  // acceso a la textura para obtener dos valores de color RGBA
  fragmentColor = mix (texture(myCubeMapTexture, RefractDir),
                       texture(myCubeMapTexture, ReflectDir),
                       material.refractionFactor);
}
```

Video ejemplo

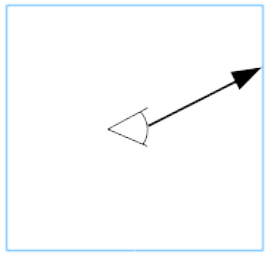


Skyboxes

Objetivo

- Representar el fondo de una escena. Contiene elementos muy distantes al observador, típicamente el sol, montañas, nubes, etc.
- Se utiliza un cubo muy grande, alrededor del observador.

El Shader para Skybox



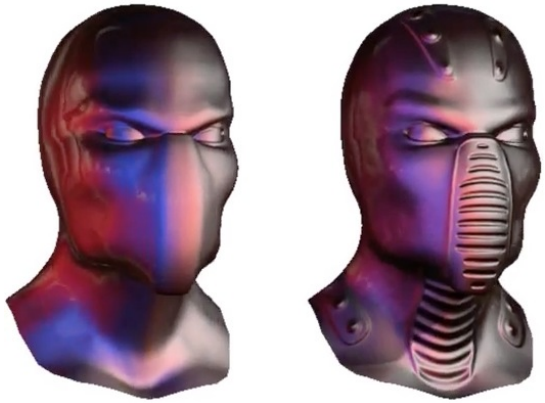
Listado 3: Shader para Skybox

```
// Vertex shader
...
in  vec3 VertexPosition;
out vec3 R;

void main()  {

  // simplemente asigna a R las coordenadas del vertice
  R = VertexPosition;
   ...
}

// Fragment shader
...
uniform samplerCube myCubeMapTexture;     // la textura
in vec3 R;

void main() {
   ...
  // acceso a la textura para obtener un valor de color RGBA
  fragmentColor = texture(myCubeMapTexture, R);
}
```

Video ejemplo

Hoy veremos...

Normal Mapping



Descripción

- Consiste en modificar la normal de la superficie para dar la ilusión de rugosidad.
- La geometría es la misma.
- Las normales modificadas se precalculan y se almacenan en un *Normal map o Bump map*.
- El *Normal map* se porporciona a la GPU como textura y contiene valores de color.

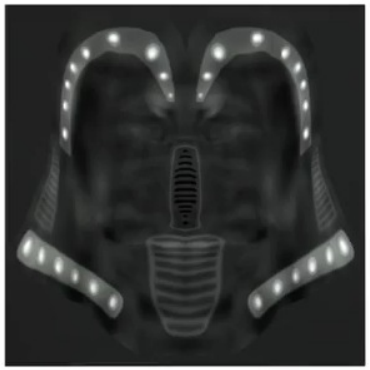
Normal Mapping

¿Cómo operamos?

- El cálculo de la iluminación se realiza en el espacio de la tangente.
- Necesitas conocer el vector tangente para cada vértice.
- En el procesador de vértices:
  - El tercer vector, llamado binormal, lo obtienes mediante el producto vectorial de la normal y la tangente.
  - Con los tres vectores creas la matriz que realiza el cambio de base de manera que la normal coincida con el Z, la tangente con el X y la binormal con el Y.
  - Operas el vector L y el vector V por esta matriz, y los resultados los interpolas para cada fragmento.
- En el procesador de fragmentos:
  - Obtienes la normal, N, del mapa de normales.
  - Y con N, L, y V aplicas el modelo de iluminación.

Hoy veremos...

Displacement Mapping



Descripción

- Consiste en aplicar un desplazamiento en cada vértice.
- El caso más sencillo es aplicarlo en la dirección de la normal.
- El desplazamiento se puede almacenar en una textura a la que se le conoce como mapa de desplazamiento.
- Desde el *vertex shader* se accede a la textura y se modifica la posición del vértice sumándole el resultado de multiplicar el despazamiento por la normal en el vértice.

Otro Ejemplo

Video ejemplo

Hoy veremos...

Alpha Mapping

Descripción

- Consiste en utilizar una textura para determinar qué partes son visibles.
- Como una plantilla por ejemplo.
- Para cada fragmento se accede a la textura y el valor devuelto te indicará si el fragmento debe continuar o no, por ejemplo.

Ejemplos