PROGRAMACIÓN 3D

Máster en Programación de Videojuegos

TEMA 9: CubeMap

Juan Mira Núñez

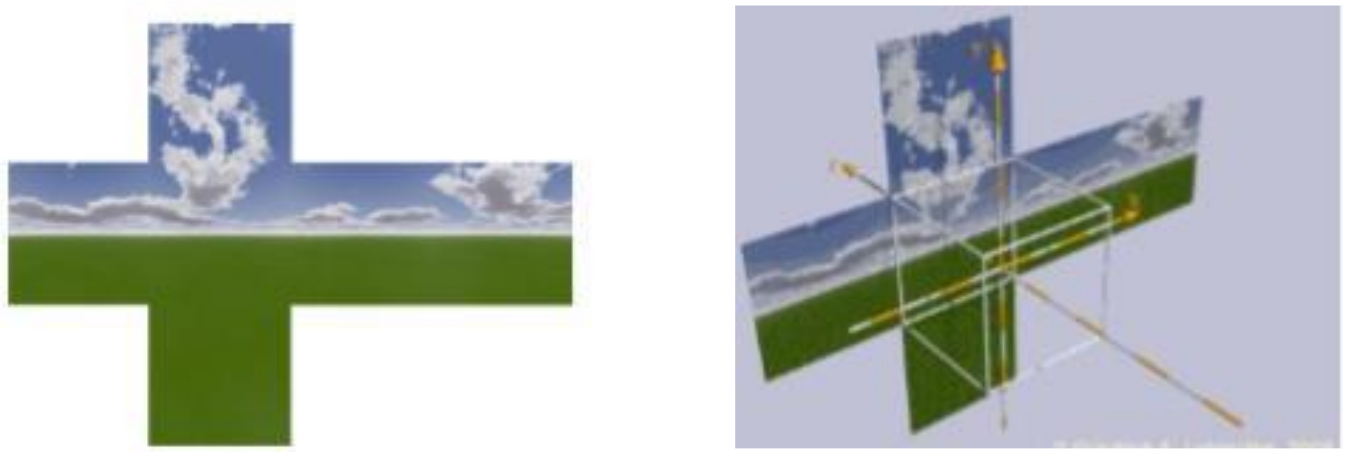
U-tad
CENTRO UNIVERSITARIO
DE TECNOLOGÍA Y ARTE DIGITAL

Índice

- Introducción
- CubeMap
- Skybox
- Reflexión
- Refracción

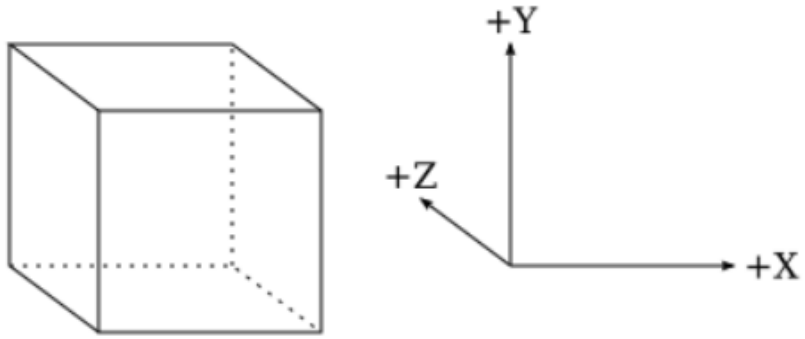
Introducción.

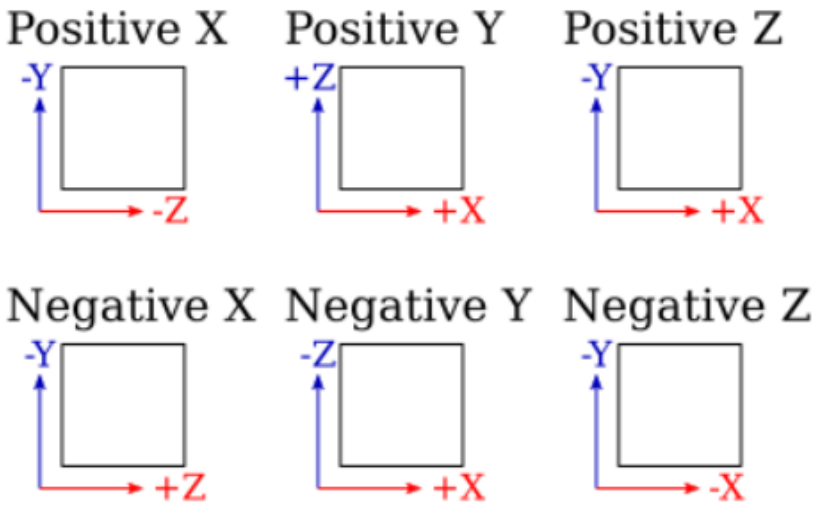
- Un cubemap es una textura que contiene 6 imágenes que representan las caras de un cubo.



- El sistema de coordenadas de texturas que se usa, es un vector de dirección que representa una dirección desde el centro del cubo al valor que queremos acceder.
- Podemos usarlo para generar un skybox , reflejos, etc.

Cargar un cubemap





Texture target	Orientation
GL_TEXTURE_CUBE_MAP_POSITIVE_X	Right
GL_TEXTURE_CUBE_MAP_NEGATIVE_X	Left
GL_TEXTURE_CUBE_MAP_POSITIVE_Y	Top
GL_TEXTURE_CUBE_MAP_NEGATIVE_Y	Bottom
GL_TEXTURE_CUBE_MAP_POSITIVE_Z	Back
GL_TEXTURE_CUBE_MAP_NEGATIVE_Z	Front

Cargar un cubemap

1. Para cargar un cubemap , hacemos como con el resto de texturas.

glGenTextures (1, &textID);

glBindTexture (GL_TEXTURE_CUBE_MAP , textID);

2. Ahora, necesitamos llamar 6 veces a glTextImage2D para setear los datos de cada textura en una cara del cubo.

glTexImage2D(GL_TEXTURE_CUBE_MAP_POSITIVE_X, 0, GL_RGB, width , height , 0, GL_RGB, GL_UNSIGNED_BYTE, data);

glTexImage2D(GL_TEXTURE_CUBE_MAP_NEGATIVE_X, 0, GL_RGB, width , height , 0, GL_RGB, GL_UNSIGNED_BYTE, data);

glTexImage2D(GL_TEXTURE_CUBE_MAP_POSITIVE_Y, 0, GL_RGB, width , height , 0, GL_RGB, GL_UNSIGNED_BYTE, data);

glTexImage2D(GL_TEXTURE_CUBE_MAP_ NEGATIVE _Y, 0, GL_RGB, width , height , 0, GL_RGB, GL_UNSIGNED_BYTE, data);

glTexImage2D(GL_TEXTURE_CUBE_MAP_POSITIVE_Z, 0, GL_RGB, width , height , 0, GL_RGB, GL_UNSIGNED_BYTE, data);

glTexImage2D(GL_TEXTURE_CUBE_MAP_ NEGATIVE _Z, 0, GL_RGB, width , height , 0, GL_RGB, GL_UNSIGNED_BYTE, data);

Skybox

- Un skybox es un cubo con una textura cubemap

- Estamos “dentro” del cubo, con lo que se deberán pintar las caras interiores.

- Siempre lo renderizamos centrado en la cámara y siempre será el primer elemento a renderizar.

- Para centrarlo en la cámara, no usamos la matriz modelo y eliminamos la translación de la cámara de la matriz vista:

```
Shader->setMatrix(shader->getLocation(“SkyboxMVP"),
State::projectionMatrix * glm::mat4(glm::mat3(State::viewMatrix)));
```

Vertex shader

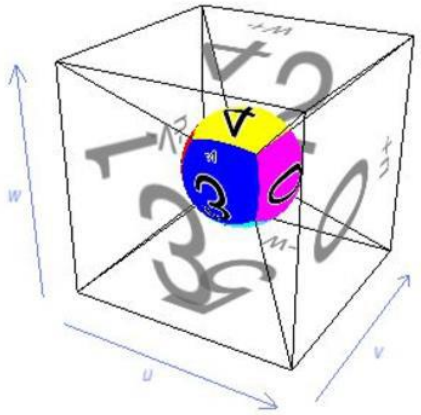
```
uniform mat4 SkyboxMVP;
attribute vec3 vpos;
varying vec3 uvw;

void main() {
        gl_Position = SkyboxMVP * vec4(vpos, 1.0);
        uvw = vpos;
}
```

Fragment shader

```
varying vec3 uvw;
uniform samplerCube texMap;


void main()
{
	gl_FragColor = texture(texMap, normalize(uvw));
}
```

Reflexión y Refracción

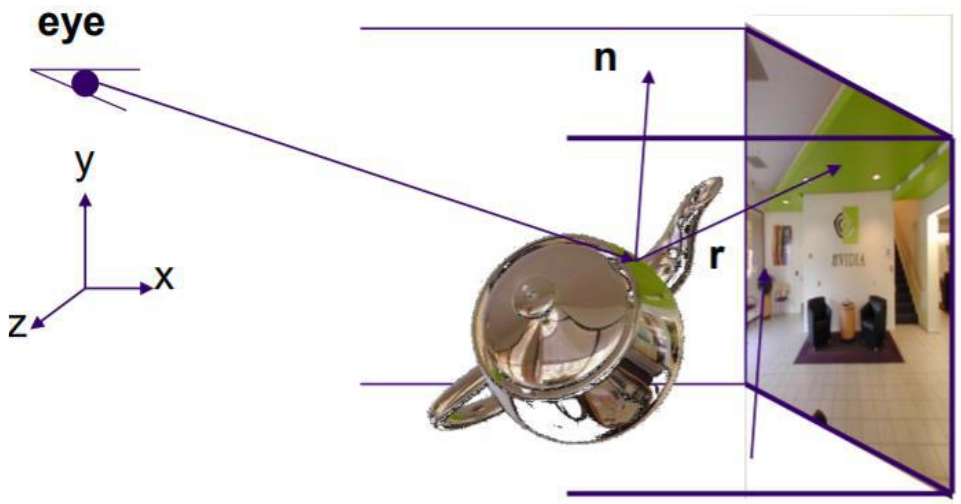
Imaginemos que queremos reflejar el entorno en un objeto.



1. Situamos la tetera en el centro del cubo.
2. Por cada vértice de la tetera, proyectamos su vector dirección al cubo (por ejemplo, el vector reflejado desde la vista a la normal al vértice).
3. Cuando el vector intersecta con el cubo, el fragmento de textura en ese punto, es el que se aplica al objeto.

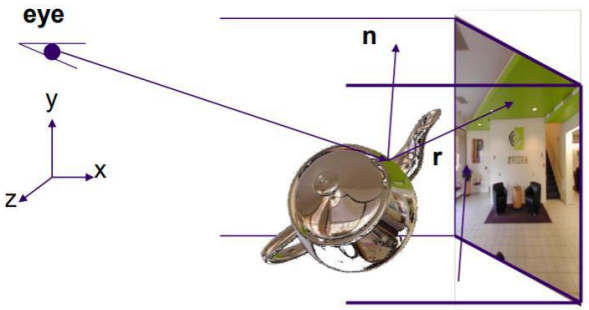
Reflexión

- Vector Reflejado:




```
vec3 eye = normalize(vec3(ModelMatrix * vpos , 1)) – eyePos;
vec3 normal = vec3(ModelMatrix * vec4(vnormal, 0));
vec3 R = normalize(reflect(eye, normal);
```

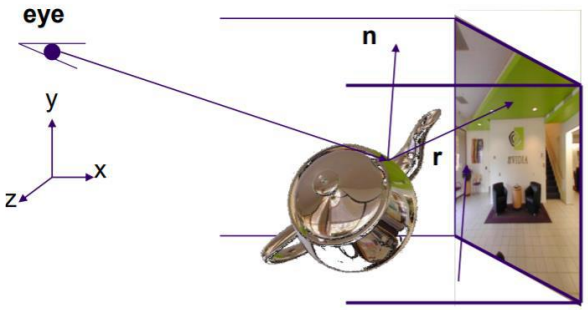
Reflexión Vertex Shader



- Vector Reflejado:

```
out vec3 R;
in vec3 vpos;
in vec3 vnormal;
uniform mat4 ModelMatrix;
uniform mat4 ViewMatrix;
uniform mat4 ProjectionMatrix;

void main (){
        gl_Position = ProjectionMatrix * ViewMatrix * ModelMatrix * vpos;
        vec3 eye = normalize(vec3(ModelMatrix * vec4(vpos, 1)) – eyePos;
        vec3 normal = vec3(ModelMatrix * vec4(vnormal, 0));
        vec3 R = normalize(reflect(eye.xyz, normal);
}
```

Reflexión Fragment Shader



- Vector Reflejado:

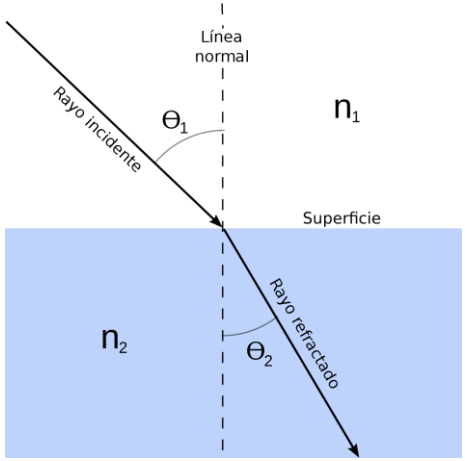
```
in vec3 R;
uniform samplerCube reflectionMap;


void main()
{
        gl_FragColor = texture(reflectionMap, R);
}
```

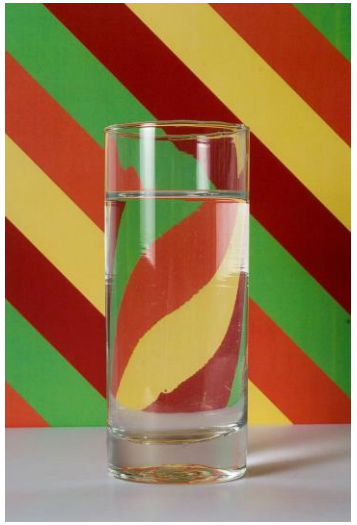
Refracción

La refracción es el cambio de dirección y velocidad que experimenta una onda de luz al pasar de un medio a otro, ya sea líquido o gaseoso, con distinto índice refractivo.

Solo se produce si la onda incide oblicuamente sobre la superficie de separación de los dos medios y si estos tienen índices de refracción distintos.

La refracción se origina en el cambio de velocidad de propagación de la onda señalada.





Refracción

El índice de refracción de un medio es una medida que indica cuánto se reduce la velocidad de la luz (o de otras ondas tales como ondas acústicas) dentro del medio.

Refracción

También se puede refractar el entorno:

```
vec3 eye = normalize(vec3(ModelMatrix * vec4(vpos , 1)) – eyePos;
vec3 normal = vec3(ModelMatrix * vec4(vnormal, 0));
vec3 R = normalize(refract(eye.xyz, normal, 0.95);
```

¿Dudas?