Texture Coordinates

Dinesh K. Pai

Textbook Chapter 15

Some slides courtesy of M. Kim, KAIST

1

Today

- Reminders:
  - Assignment 3 due today
  - Please sign up for face-to-face grading
- Texturing in Three.js
- Cube maps
- Projector maps

2

Texture mapping in Three.js

- Interactive demo, Earth texture mapping

```
var earthColorTexture =
        new THREE.ImageUtils.loadTexture('images/earthmap1k.jpg');

var earthBumpTexture =
        new THREE.ImageUtils.loadTexture('images/earthbump1k.jpg');

var earthMaterial = new THREE.MeshPhongMaterial(
    {
        map: earthColorTexture,
        bumpMap: earthBumpTexture
    } );

var gem = new THREE.Mesh(gemGeometry, earthMaterial);
```

3

Generating your own Texture Coordinates

- Can be done in Maya, 3DS Max, Blender and other 3D modeling software. This is how it's done in production applications
- Legacy OpenGL had a function (glTexGen) to do this, removed from current versions
- In production, coordinates are designed with model (or "painted" on 3D model)
- Useful texture coordinates can often be computed in shaders (e.g., projection, environment maps)

Environment cube maps

- Textures can also be used to model the environment in the distance around the object being rendered.
- In this case, we typically use 6 square textures representing the faces of a large cube surrounding the scene.

5

Environment cube maps

- Each texture pixel represents the color as seen along one direction in the environment.
- This is called a *cube map*. GLSL provides a cube-texture data type, samplerCube, specifically for this purpose.

6

Environment cube maps

- During the shading of a point, we can treat the material at that point as a perfect mirror and fetch the environment data from the appropriate incoming direction.

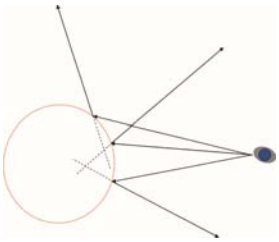
7

Environment map shader

- We calculated $B(\vec{v})$ in a previous lecture.
- This bounced vector will point points towards the environment direction, which would be observed in a mirrored surface.
- By looking up the cube map, using this direction, we give the surface the appearance of a mirror.

8

Geometry of Cube Mapping



9

Environment map shader

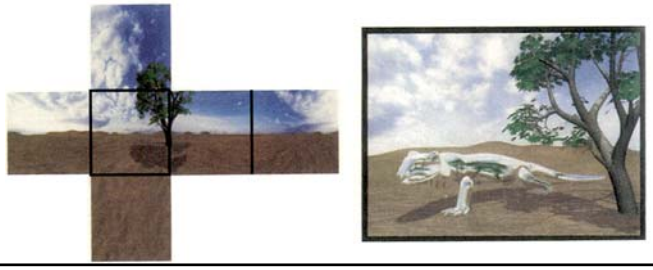
- Fragment shader

```
#version 330
uniform samplerCube uTexUnit0;
in vec3 vNormal;
in vec4 vPosition;
out vec4 fragColor;

vec3 reflect(vec3 w, vec3 n){
   return n*(dot(w,n)*2.0) - w; // bounce vector
}

void main() {
   vec3 normal = normalize(vNormal);
   vec3 reflected = reflect(normalize(vec3(-vPosition)), normal);
   vec4 texColor0 = textureCube(uTexUnit0, reflected);
   fragColor = vec4(texColor0.r, texColor0.g, texColor0.b, 1.0);;
}
```

10

Environment map shader

- -vPosition represents the view vector $\vec{v}$
- textureCube is a special GLSL function that takes a direction vector and returns the color stored at this direction in the cube texture map.
- Here we assume eye-coordinates, but frame changes may be needed.



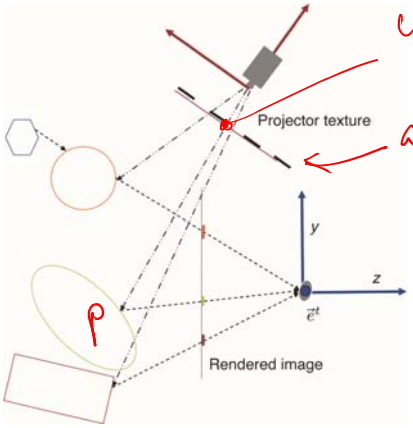
11

Projector texture mapping

- There are times when we wish to glue our texture onto our triangles using a *projector* model, instead of the affine gluing model.
- For example, we may wish to simulate a slide projector illuminating some triangles in space.

12

Geometry of Projector Textures



u,v coordinate of p in slide tex.

a "slide" being projected

13