Lab 09: Bump Mapping

CS423: Computer Graphics

Contents

1 Overview

2 Instructions

Let's start with the basic HTML we need to use:

```
<!DOCTYPE html>

<html>

<head>
    <title>Example 10.03 - Normal maps</title>
    <script type="text/javascript" src="../libs/three.js"></script>

    <script type="text/javascript" src="../libs/stats.min.js"></script>
    <script type="text/javascript" src="../libs/dat.gui.min.js"></script>
    <style>
        body {
            /* set margin to 0 and overflow to hidden, to go fullscreen */
            margin: 0;
            overflow: hidden;
        }
    </style>
</head>
<body>

<div id="Stats-output">
</div>
<!-- Div which will hold the Output -->
<div id="WebGL-output">
</div>

<!-- Javascript code that runs our Three.js examples -->
<script type="text/javascript" src="04-normal-map.js"> </script>
</body>
</html>
```

Now we do the Javascript we need to put in to make this work. Let's set the scene by putting the following into `04-basic-material.js`:

```
    function init() {

        var stats = initStats();
```

```
4
        // create a scene, that will hold all our elements such as objects, cameras and lights.
6       var scene = new THREE.Scene();

8       // create a camera, which defines where we're looking at.
        var camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1,
             1000);
10
        // create a render and set the size
12      var webGLRenderer = new THREE.WebGLRenderer();
        webGLRenderer.setClearColor(new THREE.Color(0xEEEEEE, 1.0));
14      webGLRenderer.setSize(window.innerWidth, window.innerHeight);
        webGLRenderer.shadowMapEnabled = true;
16
        var sphere1 = createMesh(new THREE.BoxGeometry(15, 15, 15), "plaster.jpg");
18      sphere1.rotation.y = -0.5;
        sphere1.position.x = 12;
20      scene.add(sphere1);

22      var sphere2 = createMesh(new THREE.BoxGeometry(15, 15, 15), "plaster.jpg", "plaster-
            normal.jpg");
        sphere2.rotation.y = 0.5;
24      sphere2.position.x = -12;
        scene.add(sphere2);
26      console.log(sphere2.geometry.faceVertexUvs);

28      var floorTex = THREE.ImageUtils.loadTexture("../assets/textures/general/floor-wood.jpg");
        var plane = new THREE.Mesh(new THREE.BoxGeometry(200, 100, 0.1, 30), new THREE.
            MeshPhongMaterial({
30          color: 0x3c3c3c,
            map: floorTex
32      }));
        plane.position.y = -7.5;
34      plane.rotation.x = -0.5 * Math.PI;
        scene.add(plane);
36
        // position and point the camera to the center of the scene
38      camera.position.x = 00;
        camera.position.y = 12;
40      camera.position.z = 38;
        camera.lookAt(new THREE.Vector3(0, 0, 0));
42
        var ambiLight = new THREE.AmbientLight(0x242424);
44      scene.add(ambiLight);

46      var light = new THREE.SpotLight();
        light.position.set(0, 30, 30);
48      light.intensity = 1.2;
        scene.add(light);
50


52      var pointColor = "#ff5808";
        var directionalLight = new THREE.PointLight(pointColor);
54
        scene.add(directionalLight);
56
        // add a small sphere simulating the pointlight
58      var sphereLight = new THREE.SphereGeometry(0.2);
        var sphereLightMaterial = new THREE.MeshBasicMaterial({color: 0xac6c25});
60      var sphereLightMesh = new THREE.Mesh(sphereLight, sphereLightMaterial);
        sphereLightMesh.castShadow = true;
62
        sphereLightMesh.position = new THREE.Vector3(3, 3, 3);
64      scene.add(sphereLightMesh);
```

```
66          // add the output of the renderer to the html element
            document.getElementById("WebGL-output").appendChild(webGLRenderer.domElement);
68
            // call the render function
70          var step = 0;

72          // setup the control gui
            var controls = new function () {
74              this.normalScale = 1;
                this.changeTexture = "plaster";
76              this.rotate = false;

78              this.changeTexture = function (e) {
                    var texture = THREE.ImageUtils.loadTexture("../assets/textures/general/" + e + ".
                        jpg");
80                  sphere2.material.map = texture;
                    sphere1.material.map = texture;
82
                    var bump = THREE.ImageUtils.loadTexture("../assets/textures/general/" + e + "-
                        normal.jpg");
84                  sphere2.material.normalMap = bump;
                };
86
                this.updateBump = function (e) {
88                  sphere2.material.normalScale.set(e, e);
                }
90          };


92
            var gui = new dat.GUI();
94          gui.add(controls, "normalScale", -2, 2).onChange(controls.updateBump);
            gui.add(controls, "changeTexture", ['plaster', 'bathroom', 'metal-floor']).onChange(
                controls.changeTexture);
96          gui.add(controls, "rotate");


98
            render();
100


102         var invert = 1;
            var phase = 0;
104
            function render() {
106             stats.update();
                step += 0.1;
108
                if (controls.rotate) {
110                 sphere1.rotation.y -= 0.01;
                    sphere2.rotation.y += 0.01;
112             }


114
                if (phase > 2 * Math.PI) {
116                 invert = invert * -1;
                    phase -= 2 * Math.PI;
118             } else {
                    phase += 0.03;
120             }

122             sphereLightMesh.position.z = +(21 * (Math.sin(phase)));
                sphereLightMesh.position.x = -14 + (14 * (Math.cos(phase)));
124
                if (invert < 0) {
126                 var pivot = 0;
                    sphereLightMesh.position.x = (invert * (sphereLightMesh.position.x - pivot)) +
```

```
                    pivot;
            }

            directionalLight.position.copy(sphereLightMesh.position);

            // render using requestAnimationFrame
            requestAnimationFrame(render);
            webGLRenderer.render(scene, camera);
        }


        function initStats() {

            var stats = new Stats();
            stats.setMode(0); // 0: fps, 1: ms

            // Align top-left
            stats.domElement.style.position = 'absolute';
            stats.domElement.style.left = '0px';
            stats.domElement.style.top = '0px';

            document.getElementById("Stats-output").appendChild(stats.domElement);

            return stats;
        }
    };

    window.onload = init;
```

Now to build a textured mesh. Note the pattern: create a loader, load a texture, assign the texture to material map, and build the mesh. Add the following code to your Javascript at the point indicated:

```
function createMesh(geom, imageFile, normal) {
    var loader = new THREE.TextureLoader();
    if (normal) {
        var t = THREE.ImageUtils.loadTexture(
                    "../assets/textures/general/" + imageFile);
        var m = THREE.ImageUtils.loadTexture(
                    "../assets/textures/general/" + normal);
        var mat2 = new THREE.MeshPhongMaterial();
        mat2.map = t;
        mat2.normalMap = m;
        var mesh = new THREE.Mesh(geom, mat2);
    } else {
        var t = THREE.ImageUtils.loadTexture(
                    "../assets/textures/general/" + imageFile);
        var mat1 = new THREE.MeshPhongMaterial({
            map: t
        });
        var mesh = new THREE.Mesh(geom, mat1);
    }
    return mesh;
}

function createNormalmapShaderMaterial(diffuseMap, normalMap) {
    var shader = THREE.ShaderLib["normalmap"];
    var uniforms = THREE.UniformsUtils.clone(shader.uniforms);
    var dT = THREE.ImageUtils.loadTexture(diffuseMap);
    var nT = THREE.ImageUtils.loadTexture(normalMap);
    uniforms["uShininess"].value = 50;
    uniforms["enableDiffuse"].value = true;
    uniforms["uDiffuseColor"].value.setHex(0xffffff);
    uniforms["tDiffuse"].value = dT;
    uniforms["tNormal"].value = nT;
```

```
    uniforms["uNormalScale"].value.set(1, 1);
    uniforms["uSpecularColor"].value.setHex(0xffffff);
    uniforms["enableSpecular"].value = true;

    return new THREE.ShaderMaterial(
            {
                fragmentShader: shader.fragmentShader,
                vertexShader: shader.vertexShader,
                uniforms: uniforms,
                lights: true
            });
}
```

Now we use this function to add objects. Add the following code where indicated:

```
var sphere1 = createMesh(new THREE.BoxGeometry(15, 15, 15), "plaster.jpg");
sphere1.rotation.y = -0.5;
sphere1.position.x = 12;
scene.add(sphere1);

var sphere2 = createMesh(new THREE.BoxGeometry(15, 15, 15),
                         "plaster.jpg",
                         "plaster-normal.jpg");
sphere2.rotation.y = 0.5;
sphere2.position.x = -12;
scene.add(sphere2);
console.log(sphere2.geometry.faceVertexUvs);

var floorTex =
  THREE.ImageUtils.loadTexture("../assets/textures/general/floor-wood.jpg");
var plane = new THREE.Mesh(new THREE.BoxGeometry(200, 100, 0.1, 30),
                           new THREE.MeshPhongMaterial({
                           color: 0x3c3c3c,
                           map: floorTex
                           }));
plane.position.y = -7.5;
plane.rotation.x = -0.5 * Math.PI;
scene.add(plane);
```

And then add a function to the do the rendering:

```
step = 0;
function render() {
    stats.update();
    step += 0.1;

    if (controls.rotate) {
        sphere1.rotation.y -= 0.01;
        sphere2.rotation.y += 0.01;
    }


    if (phase > 2 * Math.PI) {
        invert = invert * -1;
        phase -= 2 * Math.PI;
    } else {
        phase += 0.03;
    }

    sphereLightMesh.position.z = +(21 * (Math.sin(phase)));
    sphereLightMesh.position.x = -14 + (14 * (Math.cos(phase)));

```

```
    if (invert < 0) {
        var pivot = 0;
        sphereLightMesh.position.x =
            (invert * (sphereLightMesh.position.x - pivot)) + pivot;
    }
    directionalLight.position.copy(sphereLightMesh.position);
    // render using requestAnimationFrame
    requestAnimationFrame(render);
    webGLRenderer.render(scene, camera);
}
```

And let's add the controls for the GUI:

```
var controls = new function () {
    this.normalScale = 1;
    this.changeTexture = "plaster";
    this.rotate = false;
    this.changeTexture = function (e) {
        var texture =
          THREE.ImageUtils.loadTexture("../assets/textures/general/" +
                  e +
                  ".jpg");
        sphere2.material.map = texture;
        sphere1.material.map = texture;
        var bump =
           THREE.ImageUtils.loadTexture("../assets/textures/general/" +
                   e +
                   "-normal.jpg");
        sphere2.material.normalMap = bump;
    };
    this.updateBump = function (e) {
        sphere2.material.normalScale.set(e, e);
    }
};
```

Save both files and check out the result.

3 Submission instructions

Please create a PDF file with the following:

- A screen-shot of both your webapps displayed in the browser.
- HTML and JS files for each webapp

Attach this PDF file to the submission link in Blackboard.