Lab 06: Materials

CS423: Computer Graphics

Contents

1 Overview

2 Instructions

We'll look at the behavior of each of the base material types: `MeshBasicMaterial`, `MeshLambertMaterial`, and `MeshPhongMaterial`.

2.1 `MeshBasicMaterial`

Start with the following HTML document (with the name `03-basic-mesh-material.html`):

```
<!DOCTYPE html>

<html>

<head>
    <title>Example 03.03 - MeshBasicMaterial</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="03-basic-mesh-material.js">
```

```
</script>
</body>
</html>
```

Now let's setup the base scene we want to display:

```
// once everything is loaded, we run our Three.js stuff.
function init() {

    var stats = initStats();

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

    // create a camera, which defines where we're looking at.
    var camera = new THREE.PerspectiveCamera(45,
            window.innerWidth / window.innerHeight,
            0.1,
            1000);
    // create a render and set the size
    var renderer;
    var renderer = new THREE.WebGLRenderer();
    renderer.setClearColor(new THREE.Color(0xEEEEEE, 1.0));
    renderer.setSize(window.innerWidth, window.innerHeight);
    renderer.shadowMap.enabled = true;

    var groundGeom = new THREE.PlaneGeometry(100, 100, 4, 4);
    var groundMesh = new THREE.Mesh(groundGeom,
                            new THREE.MeshBasicMaterial({color: 0x777777}));
    groundMesh.rotation.x = -Math.PI / 2;
    groundMesh.position.y = -20;
    scene.add(groundMesh);

    var sphereGeometry = new THREE.SphereGeometry(14, 20, 20);
    var cubeGeometry = new THREE.BoxGeometry(15, 15, 15);
    var planeGeometry = new THREE.PlaneGeometry(14, 14, 4, 4);


    var meshMaterial = new THREE.MeshBasicMaterial({color: 0x7777ff});

    var sphere = new THREE.Mesh(sphereGeometry, meshMaterial);
    var cube = new THREE.Mesh(cubeGeometry, meshMaterial);
    var plane = new THREE.Mesh(planeGeometry, meshMaterial);

    // position the sphere
    sphere.position.x = 0;
    sphere.position.y = 3;
    sphere.position.z = 2;

    cube.position = sphere.position;
    plane.position = sphere.position;

    // add the sphere to the scene
    scene.add(cube);

    // position and point the camera to the center of the scene
    camera.position.x = -20;
    camera.position.y = 50;
    camera.position.z = 40;
    camera.lookAt(new THREE.Vector3(10, 0, 0));

    // add subtle ambient lighting
```

```
    var ambientLight = new THREE.AmbientLight(0x0c0c0c);
59  scene.add(ambientLight);

61  // add spotlight for the shadows
    var spotLight = new THREE.SpotLight(0xffffff);
63  spotLight.position.set(-40, 60, -10);
    spotLight.castShadow = true;
65  scene.add(spotLight);

67  // add the output of the renderer to the html element
    document.getElementById("WebGL-output").appendChild(renderer.domElement);
69
    // call the render function
71  var step = 0;
    var oldContext = null;
73
    var controls = new function () {
75      this.rotationSpeed = 0.02;
        this.bouncingSpeed = 0.03;
77
        this.opacity = meshMaterial.opacity;
79      this.transparent = meshMaterial.transparent;
        this.visible = meshMaterial.visible;
81      this.side = "front";

83      this.color = meshMaterial.color.getStyle();
        this.wireframe = meshMaterial.wireframe;
85      this.wireframeLinewidth = meshMaterial.wireframeLinewidth;
        this.wireFrameLineJoin = meshMaterial.wireframeLinejoin;
87
        this.selectedMesh = "cube";
89


91  };

93  var gui = new dat.GUI();


95
    var spGui = gui.addFolder("Mesh");
97  spGui.add(controls, 'opacity', 0, 1).onChange(function (e) {
        meshMaterial.opacity = e
99  });
    spGui.add(controls, 'transparent').onChange(function (e) {
101     meshMaterial.transparent = e
    });
103 spGui.add(controls, 'wireframe').onChange(function (e) {
        meshMaterial.wireframe = e
105 });
    spGui.add(controls, 'wireframeLinewidth', 0, 20).onChange(function (e) {
107     meshMaterial.wireframeLinewidth = e
    });
109 spGui.add(controls, 'visible').onChange(function (e) {
        meshMaterial.visible = e
111 });
    spGui.add(controls, 'side', ["front", "back", "double"]).onChange(function (e) {
113
        switch (e) {
115     case "front":
            meshMaterial.side = THREE.FrontSide;
117         break;
        case "back":
119         meshMaterial.side = THREE.BackSide;
            break;
121     case "double":
            meshMaterial.side = THREE.DoubleSide;
```

```
            break;
        }
        meshMaterial.needsUpdate = true;
    });
    spGui.addColor(controls, 'color').onChange(function (e) {
        meshMaterial.color.setStyle(e)
    });
    spGui.add(controls, 'selectedMesh', ["cube", "sphere", "plane"]).onChange(function (e) {

        scene.remove(plane);
        scene.remove(cube);
        scene.remove(sphere);

        switch (e) {
        case "cube":
            scene.add(cube);
            break;
        case "sphere":
            scene.add(sphere);
            break;
        case "plane":
            scene.add(plane);
            break;

        }

        scene.add(e);
    });


    render();

    function render() {
        stats.update();

        cube.rotation.y = step += 0.01;
        plane.rotation.y = step;
        sphere.rotation.y = step;

        // render using requestAnimationFrame
        requestAnimationFrame(render);
        renderer.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;
```

Save load both files and load the HTML file into your browser.

2.2 Lambert Shading

Copy `03-basic-mesh-material.html` to `03-lambert-mesh-material.html`. And change the name of the Javascript file to `03-lambert-mesh-material.js`.

Change the material for the object being rendered from `MeshBasicMaterial` to `MeshLambertMaterial`.
Save and load the HTML file into your browser.

2.3 Phong Shading

Copy `03-basic-mesh-material.html` to `03-phong-mesh-material.html`. And change the name of the Javascript file to `03-phong-mesh-material.js`.

Change the material for the object being rendered from `MeshBasicMaterial` to `MeshPhoneMaterial`.
Save and load the HTML file into your browser.

2.4 Your Turn: A Few Experiments

Try the following:

1. Adjust the color of the object being rendered.
2. Change the location of the spotlight and see how the lighting changes from location to location.

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.