Lab 03: Perspective vs. Orthographic Camera

CS423: Computer Graphics

Contents

1 Overview

As you might expect from the lecture, you get very different results depending upon which camera you select in `THREE.js`. In this lab, we will consider what that looks like.

2 Instructions

Starting from our WebGL HTML template from Lab 01, create a new file named `02-both-cameras.html` with the following HTML:

```
<!DOCTYPE html>
<HTML>
<HEAD>
    <TITLE>Example 02.01 - Basic Scene</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="02-both-cameras.js">
</SCRIPT>
</BODY>
</HTML>
```

Now download the Javascript file `02-both-cameras.js` that is attached to this lab.

Now for the interesting stuff. Open the Javascript file and look at the code in the `init()` function. You see it follow the standard workflow: (1) create a scene, (2) add a camera, (3) place objects in the scene, and (4) render.

In this case, we construct a base plane and then use a nested loop to build a collection of cubes on that that plane.

Note that we're using the `MeshLambertMaterial` for both plane and cubes and we add both directional and ambient lighting to the scene. More on that when we turn our attention to how lighting effects are done. For now, the lighting is arranged to get us the best demo of camera effects.

2.1 Let's Update the controls

We want to be able to render this scene using either a perspective camera or a orthographic camera. We do that by adding some UI chrome to the page using `dat.gui`. Add the following to your `init()` function right before you set the camera "look at" attribute:

```
var controls = new function () {
    this.perspective = "Perspective";
    this.switchCamera = function () {
        if (camera instanceof THREE.PerspectiveCamera) {
            camera = new THREE.OrthographicCamera(window.innerWidth / -16, window.innerWidth
                / 16, window.innerHeight / 16, window.innerHeight / -16, -200, 500);
            camera.position.x = 120;
            camera.position.y = 60;
            camera.position.z = 180;
            camera.lookAt(scene.position);
            this.perspective = "Orthographic";
        } else {
            camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight,
                0.1, 1000);
            camera.position.x = 120;
            camera.position.y = 60;
            camera.position.z = 180;

            camera.lookAt(scene.position);
            this.perspective = "Perspective";
        }
    };
};

var gui = new dat.GUI();
gui.add(controls, 'switchCamera');
gui.add(controls, 'perspective').listen();
```

This code gives the user the option of selecting between cameras of each type positioned in the same location in object space. Save both your HTML and JS files. Load them into the browser and note what happens when you switch between cameras. Can you explain what's happening?

3 Submission instructions

Please create a PDF file with the following:

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

Attach this PDF file to the submission link in Blackboard.