Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа № 13

по дисциплине "Программирование графических приложений"

ТЕМА РАБОТЫ:

Поверхности вращения и параметрические параметры

Выполнил:

студент гр. ПРИм-124

Парахин К.В.

Принял:

Жигалов И.Е.

Владимир 2024 г.

Цель работы:

Изучение методов работы с поверхностями вращения и параметрическими поверхностями в WebGL.

Выполнение работы:

На основе приведенных примеров программ дополнить сцену, построенную в результате выполнения лабораторной работы по теме 9, поверхностью вращения и параметрической поверхностью в соответствии с тематикой сцены. К одной из добавленных поверхностей применить материал, к другой текстуру

Для выполнения задания возьмем объекты со сцены с лабораторной работы 9 – а именно рыбок – и добавь на эту сцену поверхность вращения (например, напоминающую по виду водоросль) и параметрическую поверхность  
  
На одной сцене все вместе у меня расположить не получилось – из-за того, что сцены с поверхностью вращения и параметрической поверхностью в приведенных примерах используют разные версии библиотек three\_min.js – и вместе все отображается некорректно  
  
Поэтому сначала объединил старые объекты рыбок с поверхностью вращения  
  
Листинг кода:  
  
var scene, camera, renderer, controls, light, lathe, pyramid1, sphere1, pyramid2, sphere2;

window.onload = function() {

    init();

    animate();

};

function init() {

    scene = new THREE.Scene();

    AddCamera(0, 300, 500);

    AddLight(0, 300, 500);

    renderer = new THREE.WebGLRenderer({ antialias: true });

    renderer.setClearColor(0x87ceeb);

    renderer.setSize(window.innerWidth, window.innerHeight);

    document.body.appendChild(renderer.domElement);

    var material = new THREE.MeshPhongMaterial({

        color: 0x32cd32,

        specular: 0x00b2fc,

        shininess: 50,

        side: THREE.DoubleSide

    });

    var points = [];

    for (var i = 0; i < 10; i += 0.5) {

        var x = Math.sin(i \* 0.5) \* 10 + 5;

        var y = i \* 10;

        points.push(new THREE.Vector3(x, y, 0));

    }

    var latheGeometry = new THREE.LatheGeometry(points, 64);

    lathe = new THREE.Mesh(latheGeometry, material);

    lathe.position.set(0, -50, 0);

    scene.add(lathe);

    var obj\_material1 = [

        new THREE.MeshPhongMaterial({ wireframe: false, side: THREE.DoubleSide, color: 0xff0000 })

    ];

    var obj\_material2 = [

        new THREE.MeshPhongMaterial({ wireframe: false, side: THREE.DoubleSide, color: 0xaacb00 })

    ];

    var radius\_top = 0;

    var radius\_bottom = 16;

    var height = 20;

    var segments = 3;

    var pyramid\_geometry = new THREE.CylinderGeometry(radius\_top, radius\_bottom, height, segments);

    var sphere\_geometry = new THREE.SphereGeometry(20, 6, 8);

    // Рыбка 1

    pyramid1 = new THREE.Mesh(pyramid\_geometry, obj\_material1[0]);

    pyramid1.position.set(-80, 60, 0);

    pyramid1.rotateY(-Math.PI / 2);

    pyramid1.translateY(20);

    pyramid1.translateX(-10);

    sphere1 = new THREE.Mesh(sphere\_geometry, obj\_material1[0]);

    sphere1.position.set(-50, 60, 0);

    sphere1.rotateY(-Math.PI / 2);

    // Рыбка 2

    pyramid2 = new THREE.Mesh(pyramid\_geometry, obj\_material2[0]);

    pyramid2.position.set(-80, 0, 0);

    pyramid2.rotateY(-Math.PI / 2);

    pyramid2.translateY(20);

    pyramid2.translateX(-10);

    sphere2 = new THREE.Mesh(sphere\_geometry, obj\_material2[0]);

    sphere2.position.set(-50, 0, 0);

    sphere2.rotateY(-Math.PI / 2);

    var light = new THREE.HemisphereLight(0x666666, 0xff0000, 10);

    scene.add(light);

    scene.add(pyramid1, sphere1, pyramid2, sphere2);

}

function animate() {

    pyramid1.rotateY(Math.PI / 360);

    sphere1.rotateY(Math.PI / 360);

    pyramid2.rotateY(Math.PI / 360);

    sphere2.rotateY(Math.PI / 360);

    lathe.rotateY(Math.PI / 360);

    renderer.render(scene, camera);

    requestAnimationFrame(animate);

}

function AddCamera(X, Y, Z) {

    camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 1, 10000);

    camera.position.set(X, Y, Z);

    controls = new THREE.TrackballControls(camera, document.body);

    controls.rotateSpeed = 2;

    controls.noZoom = false;

    controls.zoomSpeed = 1.2;

    controls.staticMoving = true;

}

function AddLight(X, Y, Z) {

    light = new THREE.DirectionalLight(0xffffff);

    light.position.set(X, Y, Z);

    scene.add(light);

}

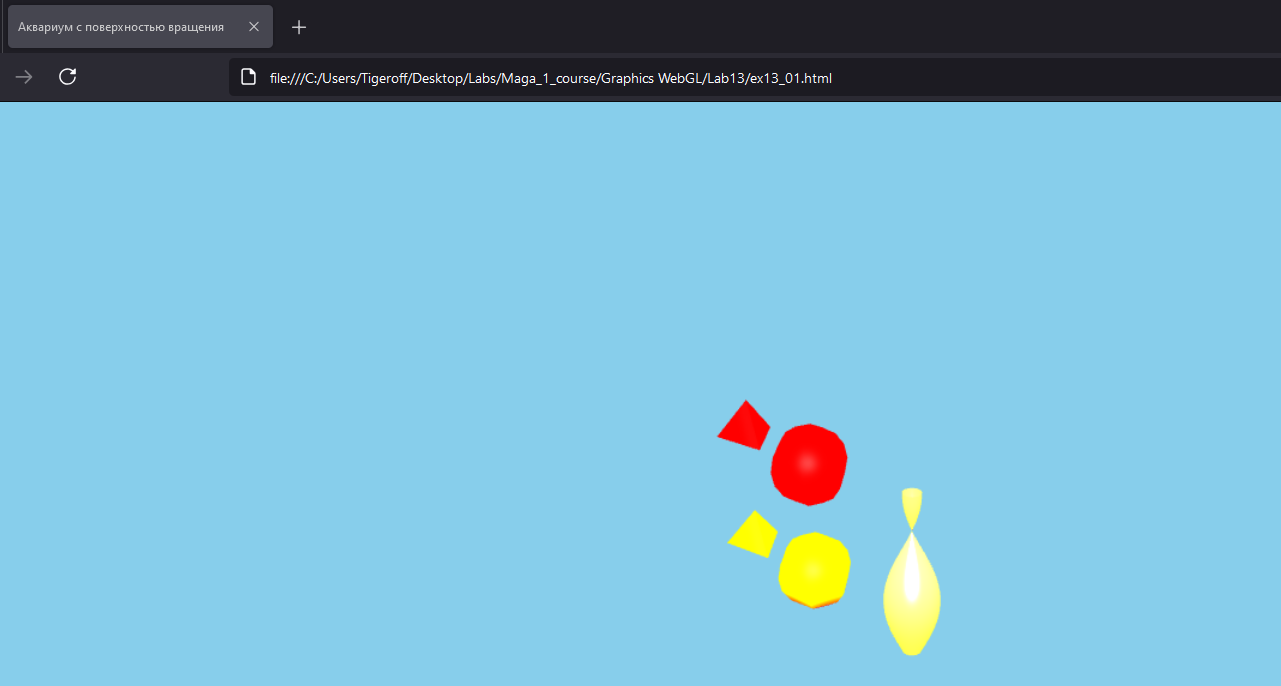


Рисунок 1. Аквариум с поверхностью вращения

Далее на отдельной сцене добавил параметрическую поверхность, напоминающую растительный объект аквариума

Листинг кода:

var container, camera, controls, scene, renderer, light;

var segments = 32;

var graphGeometry;

var graphMesh;

window.onload = function() {

    init();

    animate();

}

function init() {

    scene = new THREE.Scene();

    AddCamera(300, 600, 500);

    AddLight(0, 500, 500);

    renderer = new THREE.WebGLRenderer({ antialias: true });

    renderer.setClearColor(0x87ceeb);

    renderer.setSize(window.innerWidth, window.innerHeight);

    container = document.getElementById('MyWebGLApp');

    container.appendChild(renderer.domElement);

    ParamFunction = function(rho, teta) {

        rho = 50 \* Math.sin(Math.PI \* rho);

        teta = 2 \* Math.PI \* teta;

        var x = rho \* Math.cos(teta);

        var y = rho \* Math.sin(teta);

        var z = 10 \* Math.sin(3 \* teta);

        return new THREE.Vector3(x, z, y);

    };

    graphGeometry = new THREE.ParametricGeometry(ParamFunction, segments, segments, false);

    var material = new THREE.MeshPhongMaterial({ color: 0x32cd32, side: THREE.DoubleSide });

    graphMesh = new THREE.Mesh(graphGeometry, material);

    graphMesh.doubleSided = true;

    graphMesh.position.set(0, -50, 0);

    scene.add(graphMesh);

}

function animate() {

    requestAnimationFrame(animate);

    render();

}

function render() {

    controls.update();

    renderer.render(scene, camera);

}

function AddCamera(X, Y, Z) {

    camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 1, 10000);

    camera.position.set(X, Y, Z);

    controls = new THREE.TrackballControls(camera, container);

    controls.rotateSpeed = 2;

    controls.noZoom = false;

    controls.zoomSpeed = 1.2;

    controls.staticMoving = true;

}

function AddLight(X, Y, Z) {

    light = new THREE.DirectionalLight(0xffffff);

    light.position.set(X, Y, Z);

    scene.add(light);

}

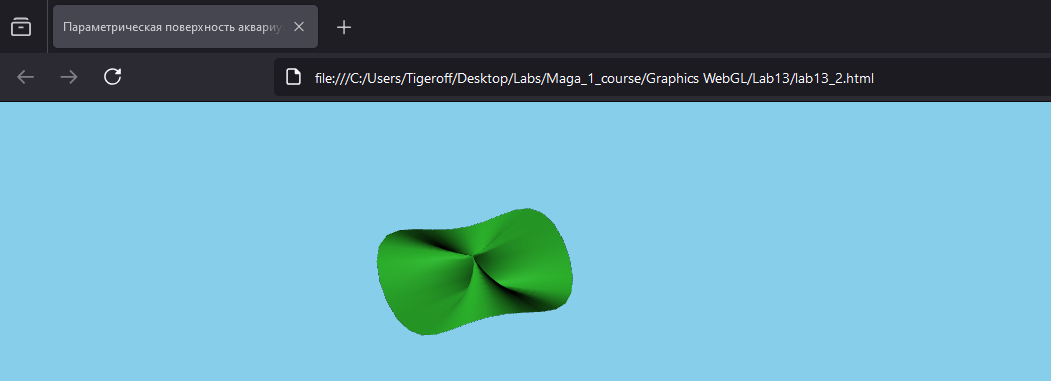


Рисунок 2. Параметрическая поверхность аквариума

Вывод

В результате выполнения работы я провел изучение методов работы с поверхностями вращения и параметрическими поверхностями в WebGL.