Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра ПМиК

Курсовая работа
по дисциплине
«Программирование мобильных устройств»

Выполнил: студент 4 курса
ИВТ, гр. ИП-813
Бурдуковский И.А.

Проверил: ассистент кафедры ПМиК
Павлова У.В.

Новосибирск 2021

Оглавление

Задание

3D графика на оценку отлично.

Студенты объединяются в группы по 3-4 человека (состав и индивидуальный объем работ обговаривается с преподавателем). Каждый участник вносит равный вклад (по объему и сложности кода) в проект. Минимум 3 различных класса или вида объектов на человека. Текстуры и звуки должны быть подобраны должным образом. Должна быть реализована возможность перемещения камеры.

Проекты:

1. Холодильник. Имеется 4 полочки холодильника, на каждой полке стоят разные продукты (разных геометрических форм: апельсин-сфера, пакет молока-пирамида, сосиски-3d овал и т.д.).

2. Вокзал. Имеется несколько ж\д составов. Состав содержит от 7 вагонов (платформы, цистерны и крытые вагоны) и может быть в движении. Дополнительно создаются различные здания и сооружения Вокзала (информационное табло, часы и.т.д., в общем, смотрите на реальный Вокзал Новосибирск Главный).

3. Супермаркет. На полочках находятся различные товары (разных геометрических форм).

4. Произвольный городской ландшафт. Например, здание ГПНТБ, перед ним фонтан, разбиты клумбы (можно добавить декоративных пирамидальных елок), автобусная остановка и т.д.

Теоретический материал

OpenGL ES 2.0.

OpenGL ES – подмножество графического интерфейса OpenGL, разработанное специально для встраиваемых систем — мобильных телефонов, карманных компьютеров, игровых консолей. OpenGL ES определяется и продвигается консорциумом Khronos Group, в который входят производители программного и аппаратного обеспечения, заинтересованные в открытом API для графики и мультимедиа.

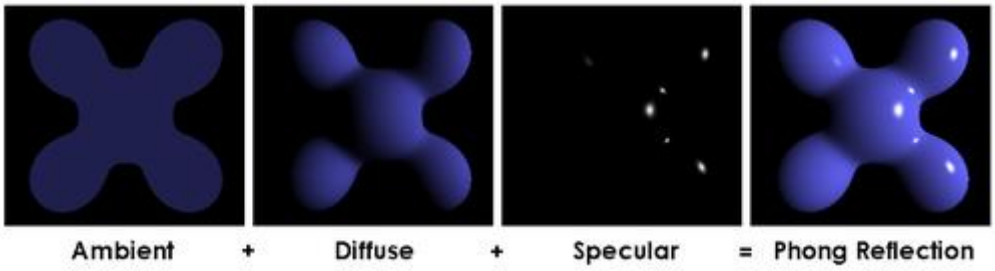
Blender

Blender – профессиональное свободное и открытое программное обеспечение для создания трёхмерной компьютерной графики, включающее в себя средства моделирования, скульптинга, анимации, симуляции, рендеринга, постобработки и монтажа видео со звуком, компоновки с помощью «узлов» (Node Compositing), а также создания 2D-анимаций. В настоящее время пользуется большой популярностью среди бесплатных 3D-редакторов в связи с его быстрым стабильным развитием и технической поддержкой.

Модель Фонга

В данной курсовой работе, было необходимо реализовать освещение по модели Фонга.

Модель Фонга - модель расчёта освещения трёхмерных объектов, в том числе полигональных моделей и примитивов, а также метод интерполяции освещения по всему объекту.



Модель Фонга рассматривает освещение исходя из трех составляющих, это: основное затенение (Ambient), диффузное рассеивание (Diffuse), блик (Specular);

Способ расчета данной модели освещения:

$$I = K_a I_a + K_d(\vec{n}, \vec{l}) + K_s(\vec{n}, \vec{h})^p$$

где

$\vec{n}$ — вектор нормали к поверхности в точке

$\vec{l}$ — падающий луч (направление на источник света)

$\vec{h}$ — отраженный луч (направление идеально отраженного от поверхности луча)

$\vec{h} = 2(\vec{l} * \vec{n})\vec{n} - \vec{l}$

K_a — коэффициент фонового освещения

K_s — коэффициент зеркального освещения

K_d — коэффициент диффузного освещения

Таким образом нам необходимо высчитать интенсивность освещения для каждой точки на поверхности сферы, а затем отобразить это через фрагментный шейдер и получить результат.

Выполнение

Для построения сложных объектов, необходимо прибегнуть к использованию сторонних программ, например, к программе Blender, при помощи 3d-редактора, мы можем создать различные объемные объекты, прилагая минимум сил, получив при этом неплохой результат.

Для курсовой работы были взяты с сайта 3д моделей следующие бесплатные объекты: продавец мороженного, тележка с мороженным, дом в азиатском стиле, котик, космические рейнджеры;

После чего, объекты были экспортированы в формат .gtlf, в данном формате, файл имеет следующий формат записи:


```
{
  "accessors": [
    {
      "bufferView": 2,
      "componentType": 5126,
      "count": 6775,
      "max": [
        52.384757995605469,
        28.862741470336914,
        55.068756103515625
      ],
      ….
      }
  ]
```

}

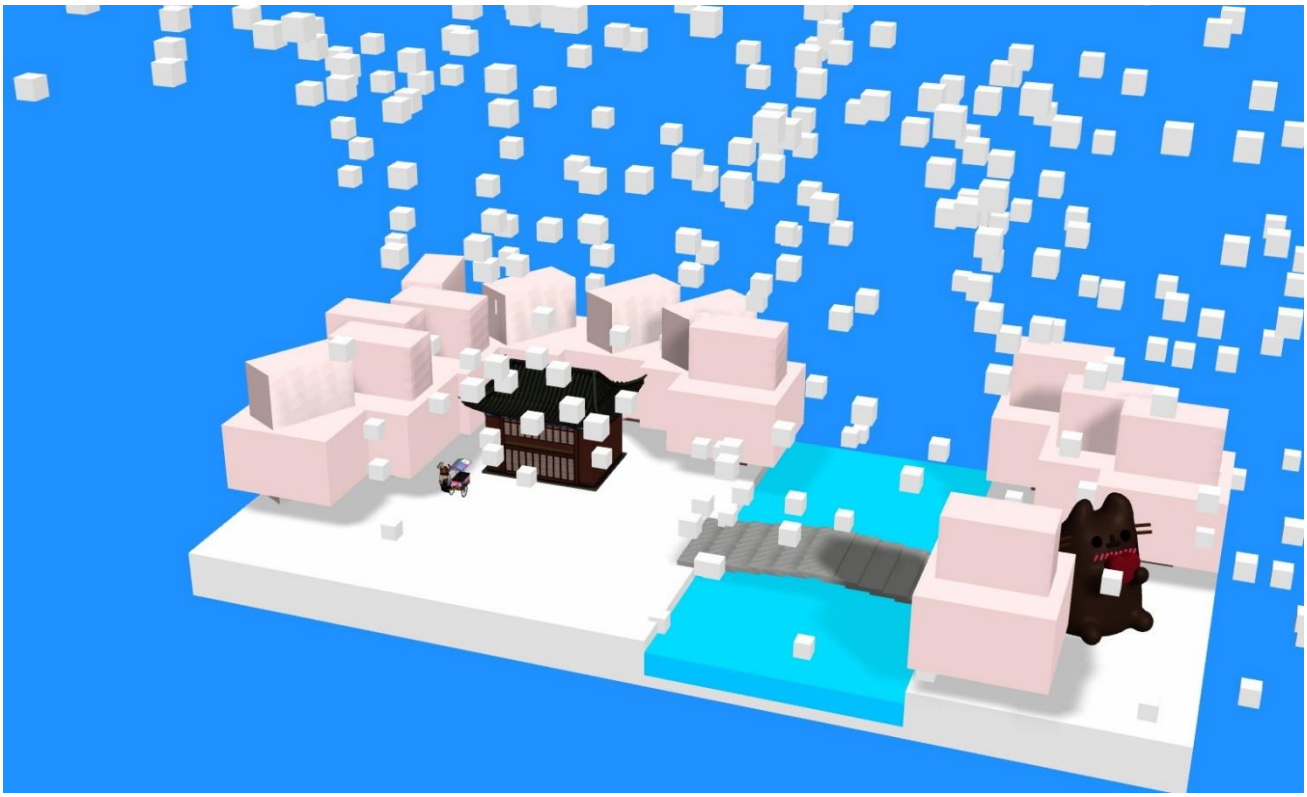
Таким образом, объект из 3д представления в программе Blender сериализуется в набор данных о его вершинах, векторах нормали и фрагментах в виде треугольников.

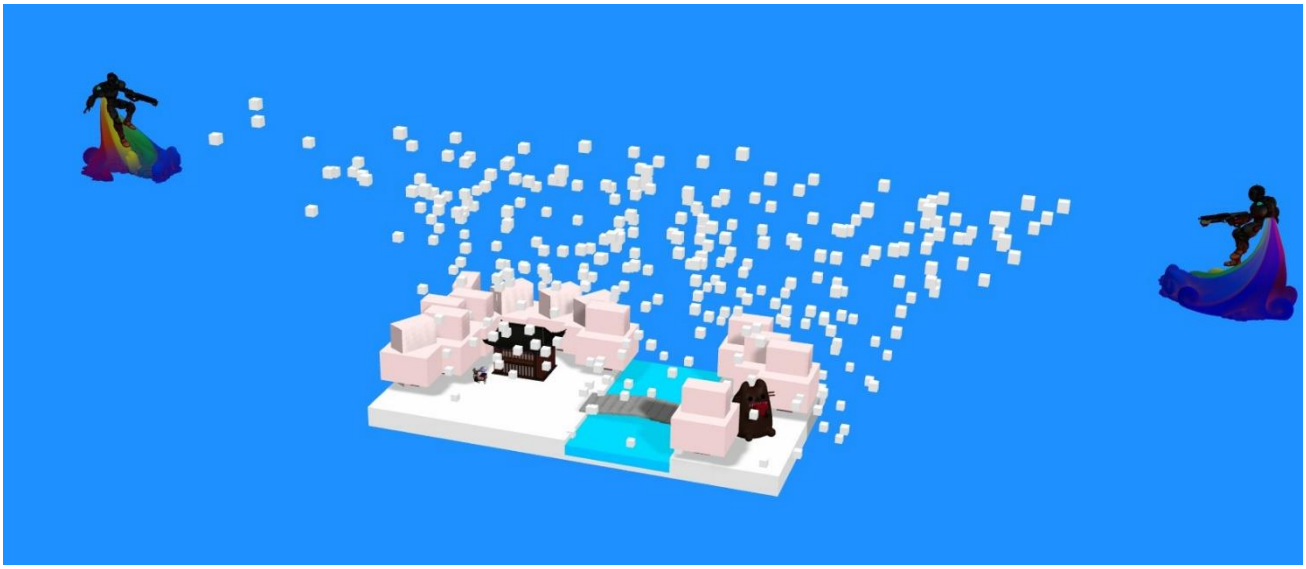
Получается мы можем заполнить буферы вершин и фрагментов десериализуя файл, и получить в памяти нашего устройства некоторый объект.

Также в коде с помощью библиотеки three js были реализованы модели деревьев, моста, самого острова с замерзшей рекой, а также снежинок.

Результат работы

В результате была получена такая сцена:





Листинг проекта

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Курсовая работа</title>
    <link rel="stylesheet" href="./style.css">

</head>
<body>

<canvas id="canvas">
<!--                style="background: url('./img/newYearRock.png')">-->
</canvas>

<script src="js/three.js"></script>
<script src="js/OrbitControls.js"></script>
<script type="module" src="./js/GLTFLoader.js"></script>

<script type="module">

    //подключение нужных библиотек
    import {OrbitControls} from
"https://threejs.org/examples/jsm/controls/OrbitControls.js";
    import {GLTFLoader} from "./js/GLTFLoader.js";
    import "./js/three.module.js";

    //задаём цвета, которые будем использовать
    var Colors = {
        cyan: 0x248079,
        brown: 0xD2691E, //ствол
        brownDark: 0x9A6169, //мостик
        green: 0x7CFC00, //для деревьев
        greenLight: 0x9ACD32, //трава
        gray: 0x888986, //камень
        pink: 0xECCDCD, // вишня
        blue: 0x00BFFF, //вода
        snow: 0xDDDDDD // снег
    };

    var scene = new THREE.Scene();//сцена

    var h = window.innerHeight, //высота
        w = window.innerWidth; //ширина

    //у камеры есть свойства
    var aspectRatio = w / h,//соотношение сторон
        fieldOfView = 25,//поле зрения
        nearPlane = .1,//плоскость отсечения
        farPlane = 1000;

    //сама камера с заданными свойстами
    var camera = new THREE.PerspectiveCamera(
        fieldOfView,
        aspectRatio,
        nearPlane,
        farPlane);
```

```
    //наш рендерер
    const renderer = new THREE.WebGLRenderer({canvas: canvas, alpha: true,
antialias: true});
    const dpi = window.devicePixelRatio;
    renderer.setSize(w * dpi, h * dpi);

    //наш canvas - берём из нашего документа тег
    const theCanvas = document.getElementById('canvas');
    theCanvas.style.width = `${w}px`;
    theCanvas.style.height = `${h}px`;

    renderer.shadowMapEnabled = true;//включаем отбрасывание теней объектами
    renderer.shadowMap.type = THREE.PCFSoftShadowMap;//Чтобы использовать этот
тип теней, вам необходимо использовать соответствующий тип карты теней
    document.body.appendChild(renderer.domElement);//присоединяем к DOM

    const controls = new OrbitControls(camera, renderer.domElement); //настройка
камеры
    camera.position.set(-5, 6, 8);//начальная позиция камеры по координатам
    camera.lookAt(new THREE.Vector3(0, 0, 0));//указываем на центр сцены

    //Добавляем освещение
    var light = new THREE.AmbientLight(0xffffff, 0.5);//добавляем окружающие
освещение (фоновое)

    var shadowLight = new THREE.DirectionalLight(0xffffff, .8);//свет, который
излучается в определенном направлении
    shadowLight.position.set(200, 200, 200);
    shadowLight.castShadow = true;//отбрасываемая тень сумеречным светом

    var backLight = new THREE.DirectionalLight(0xffffff, .2);//(рассеянное)
    backLight.position.set(-50, 200, 50);
    backLight.castShadow = true; //подсветка

    //добавляем освещение на сцену
    scene.add(backLight);
    scene.add(light);
    scene.add(shadowLight);

    // левая сторона острова
    var geometry_left = new THREE.BoxGeometry(2, .2, 2); //создание куба
    var material_ground = new THREE.MeshLambertMaterial({color:
Colors.snow});//материал для окрашивания
    var ground_left = new THREE.Mesh(geometry_left, material_ground);//сетка
    ground_left.position.set(-1, 0.1, 0); //позиция
    scene.add(ground_left);//добавляем на сцену
    customizeShadow(ground_left, .25) // mess, opacity

    //река
    var geometry_river = new THREE.BoxGeometry(1, .1, 2);//создаём куб
    var material_river = new THREE.MeshLambertMaterial({color:
Colors.blue});//материалы для окрашивания
    var river = new THREE.Mesh(geometry_river, material_river);//сетка
    river.position.set(.5, .1, 0);//позиция
    scene.add(river);//добавляем на сцену
    customizeShadow(river, .08) // mess, opacity

    //река
    var geometry_river = new THREE.BoxGeometry(1, .05, 2);//создаём куб
    var river = new THREE.Mesh(geometry_river, material_ground);//сетка
    river.position.set(.5, .025, 0);
```

```
    scene.add(river);

    //правая сторона острова
    var geometry_right = new THREE.BoxGeometry(1, .2, 2);
    var ground_right = new THREE.Mesh(geometry_right, material_ground);
    ground_right.position.set(1.5, 0.1, 0);
    scene.add(ground_right);
    customizeShadow(ground_right, .25) // mess, opacity

    var loader_china_house = new GLTFLoader();
    var obj_china_house;
    loader_china_house.load('./models/china_house/scene.gltf', function (gltf) {
        obj_china_house = gltf.scene
        obj_china_house.obj_name = "china_town";
        obj_china_house.position.y = 0.187;
        obj_china_house.position.x = -0.9;
        obj_china_house.position.z = -0.30;
        obj_china_house.scale.x = 0.15;
        obj_china_house.scale.y = 0.15;
        obj_china_house.scale.z = 0.15;
        obj_china_house.rotation.y = 3.15;
        gltf.scene.scale.multiplyScalar(0.065);
        scene.add(gltf.scene);
    }, undefined, function (error) {
        console.error(error);
    });

    // повар подвижной кухни
    var loader_chef = new GLTFLoader();
    var obj_chef;
    loader_chef.load('./models/street_chef/scene.gltf', function (gltf) {
        obj_chef = gltf.scene
        obj_chef.obj_name = "chef";
        obj_chef.position.y = 0.270;
        obj_chef.position.x = -1.25;
        obj_chef.position.z = 0.12;
        obj_chef.scale.x = 0.15;
        obj_chef.scale.y = 0.15;
        obj_chef.scale.z = 0.15;
        obj_chef.rotation.y = 1.15;
        gltf.scene.scale.multiplyScalar(0.0009);
        scene.add(gltf.scene);
    }, undefined, function (error) {
        console.error(error);
    });

    // тележка с мороженным
    var loader_ice_cream_cart = new GLTFLoader();
    var obj_ice_cream_cart;
    loader_ice_cream_cart.load('./models/ice_cream_cart/scene.gltf', function
(gltf) {
        obj_ice_cream_cart = gltf.scene
        obj_ice_cream_cart.obj_name = "ice_cream_cart";
        obj_ice_cream_cart.position.y = 0.210;
        obj_ice_cream_cart.position.x = -0.64;
        obj_ice_cream_cart.position.z = -0.225;
        obj_ice_cream_cart.scale.x = 0.15;
        obj_ice_cream_cart.scale.y = 0.15;
        obj_ice_cream_cart.scale.z = 0.15;
        obj_ice_cream_cart.rotation.y = 3.15;
        gltf.scene.scale.multiplyScalar(0.0035);
```

```
        scene.add(gltf.scene);
    }, undefined, function (error) {
        console.error(error);
    });

    // рейнджер пуляющийся снежками
    var space_ranger = function (x, y, z, rotation_angle) {
        var loader_space_ranger = new GLTFLoader();
        var obj_space_ranger;
        loader_space_ranger.load('./models/space_ranger/scene.gltf', function
(gltf) {
            obj_space_ranger = gltf.scene;
            obj_space_ranger.obj_name = "space_ranger";
            obj_space_ranger.position.x = x;
            obj_space_ranger.position.y = y;
            obj_space_ranger.position.z = z;
            obj_space_ranger.scale.x = 0.15;
            obj_space_ranger.scale.y = 0.15;
            obj_space_ranger.scale.z = 0.15;
            obj_space_ranger.rotation.y = rotation_angle;
            // gltf.castShadow = true;
            // gltf.receiveShadow = true;
            gltf.scene.scale.multiplyScalar(5);
            scene.add(gltf.scene);
        }, undefined, function (error) {
            console.error(error);
        });
    }
    space_ranger(5.5, 2, 0.12, 3.15);
    space_ranger(-5, 2, 0.12, 0);

    var loader_kitten = new GLTFLoader();
    var obj_kitten;
    loader_kitten.load('./models/kitten/scene.gltf', function (gltf) {
        obj_kitten = gltf.scene
        obj_kitten.obj_name = "china_town";
        obj_kitten.position.y = 0.187;
        obj_kitten.position.x = 1.6;
        obj_kitten.position.z = 0.3;
        obj_kitten.scale.x = 0.15;
        obj_kitten.scale.y = 0.15;
        obj_kitten.scale.z = 0.15;
        obj_kitten.rotation.y = 0.75;
        gltf.scene.scale.multiplyScalar(0.6);
        scene.add(gltf.scene);
    }, undefined, function (error) {
        console.error(error);
    });

    //создание деревьев
    var tree_oak = function (x, z, rotation_angle) {

        //ствол
        var material_trunk = new THREE.MeshLambertMaterial({color:
Colors.brownDark});
        var geometry_trunk = new THREE.BoxGeometry(.15, .15, .15);
        var trunk = new THREE.Mesh(geometry_trunk, material_trunk);
        trunk.position.set(x, .275, z);
        trunk.castShadow = true;
        trunk.receiveShadow = true;
        scene.add(trunk);
```

```
        //листья
        var geometry_leaves = new THREE.BoxGeometry(.4, .60, .20);
        geometry_leaves.rotateY(rotation_angle); // поворачиваем листву
        var material_leaves = new THREE.MeshLambertMaterial({color:
Colors.pink});
        var leaves = new THREE.Mesh(geometry_leaves, material_leaves);
        leaves.position.set(x, .2 + .15 + .4 / 2, z);
        leaves.castShadow = true;
        trunk.receiveShadow = true;


        var geometry_leaves2 = new THREE.BoxGeometry(.50, .30, .50);
        geometry_leaves.rotateY(rotation_angle); // поворачиваем листву
        var material_leaves2 = new THREE.MeshLambertMaterial({color:
Colors.pink});
        var leaves2 = new THREE.Mesh(geometry_leaves2, material_leaves2);
        leaves2.position.set(x, .1 + .15 + .4 / 2, z);

        leaves2.castShadow = true;
        trunk.receiveShadow = true;
        customizeShadow(leaves, .25) // mess, opacity
        scene.add(leaves);
        scene.add(leaves2)
    }

    //деревья на левой стороне
    tree_oak(-1.75, -.85, -0.7);
    tree_oak(-1.55, -.5, 0);
    tree_oak(-1.75, -.25, 0);
    tree_oak(-1.6, -0, 0);
    tree_oak(-1.8, .35, -1.05);
    tree_oak(-1.25, -.85, 0.5);
    tree_oak(-.75, -.925, 0.5);
    tree_oak(-.35, -.85, 0.5);
    tree_oak(-.15, -.65, 0);
    //деревья на правой стороне
    tree_oak(1.25, -.85, 0);
    tree_oak(1.25, .75, 0);
    tree_oak(1.5, -.5, 0);
    tree_oak(1.75, -.35, 0);

    //настройка тени
    function customizeShadow(t, a) { //opacity, target mesh
        var material_shadow = new THREE.ShadowMaterial({opacity: a});
        var mesh_shadow = new THREE.Mesh(t.geometry, material_shadow);
        mesh_shadow.position.set(t.position.x, t.position.y, t.position.z);
        mesh_shadow.receiveShadow = true;
        scene.add(mesh_shadow);
    }

    //материал камня
    var material_stone = new THREE.MeshPhysicalMaterial({color: Colors.gray});

    //мост
    for (var i = 0; i < 6; i++) {
        var geometry_block = new THREE.BoxGeometry(.15, .02, .4);
        var block = new THREE.Mesh(geometry_block, material_stone);
        block.position.set(0 + .1 * i, .21 + 0.01 * i, .2);
        block.castShadow = true;
        block.receiveShadow = true;
```

```
        scene.add(block);
    }
    for (var i = 4; i >= 0; i--) {
        var geometry_block = new THREE.BoxGeometry(.15, .02, .4);
        var block = new THREE.Mesh(geometry_block, material_stone);
        block.position.set(1. - .1 * i, .21 + 0.01 * i, .2);
        block.castShadow = true;
        block.receiveShadow = true;
        scene.add(block);
    }

    // var snowflake_drop_texture = new
THREE.TextureLoader().load('img/rocksee2.jpg');
    // var snowflake_drop_material = new THREE.MeshLambertMaterial({map:
snowflake_drop_texture, transparent: true});

    //материал снежинки
    var snowflake_drop_material = new THREE.MeshPhongMaterial({color:
Colors.snow});

    //наши капли
    var Snowflake = function () {
        this.geometry = new THREE.BoxGeometry(.1, .1, .1);// в форме кубиков
        this.drop = new THREE.Mesh(this.geometry, snowflake_drop_material);
//сетка
        this.drop.position.set(-4.5 + Math.random(.1, .9) * 10, 2.6, -1 +
(Math.random() * 20) * .1);// с помощью математики будем добавлять рандомно
        scene.add(this.drop);//добавляем на сцену
        this.speed = 0;//скорость изначально
        this.lifespan = (Math.random() * 50) + 20;//продолжительность жизни
        let x_size_scaler = 1, y_size_scaler = 1, z_size_scaler = 1;
        //обновление капли
        this.update = function () {
            this.speed += .001;
            this.lifespan--;
            this.drop.scale.set(x_size_scaler, y_size_scaler, z_size_scaler);

            x_size_scaler -= 0.005;
            y_size_scaler -= 0.005;
            z_size_scaler -= 0.005;
            // this.drop.scale.multiplyScalar(x_size_scaler);
            this.drop.position.x += (.5 - this.drop.position.x) / 70;
            this.drop.position.y -= this.speed;
        }
    }

    var drops = [];//массив снежинок
    var count = 0;//счётчик

    var render = function () {
        requestAnimationFrame(render);
        if (count % 10 == 0) {
            count = 0;
            for (var i = 0; i < 8; i++) {
                drops.push(new Snowflake());
            }
        }
        count++;
        for (var i = 0; i < drops.length; i++) {
            drops[i].update();
            if (drops[i].lifespan < 0) {
```

```
                scene.remove(scene.getObjectById(drops[i].drop.id));
                drops.splice(i, 1);
            }
        }
        controls.update();
        renderer.render(scene, camera);
    }
    render();
</script>

</body>
</html>
```