Министерство цифрового развития, связи и
массовых коммуникаций Российской Федерации

Сибирский государственный университет телекоммуникаций и информатики

Лабораторно-практическая работа №2

по дисциплине: Программирование трехмерной графики

Вершины, индексы, цвета и текстуры

Выполнил: Русских Екатерина
Владимировна

Группа: МИТ-22

Вариант: 1

Проверила: Шлаузер Андрей
Иванович

Новосибирск, 2024

Лабораторная работа №2

Перемещение объектов в трёхмерном пространстве

Цель работы:

- изучение способов перемещения объектов в трёхмерном пространстве с помощью Three.js;

- освоение основных методов расчёта координат объектов в трёхмерном пространстве.

Задание:

Необходимо разработать веб-приложение, представляющее собой упрощённую модель Солнечной системы. Модель должна включать следующие объекты:

- карта звёздного неба;
- Солнце;
- Меркурий;
- Венера;
- Земля и Луна
- Марс

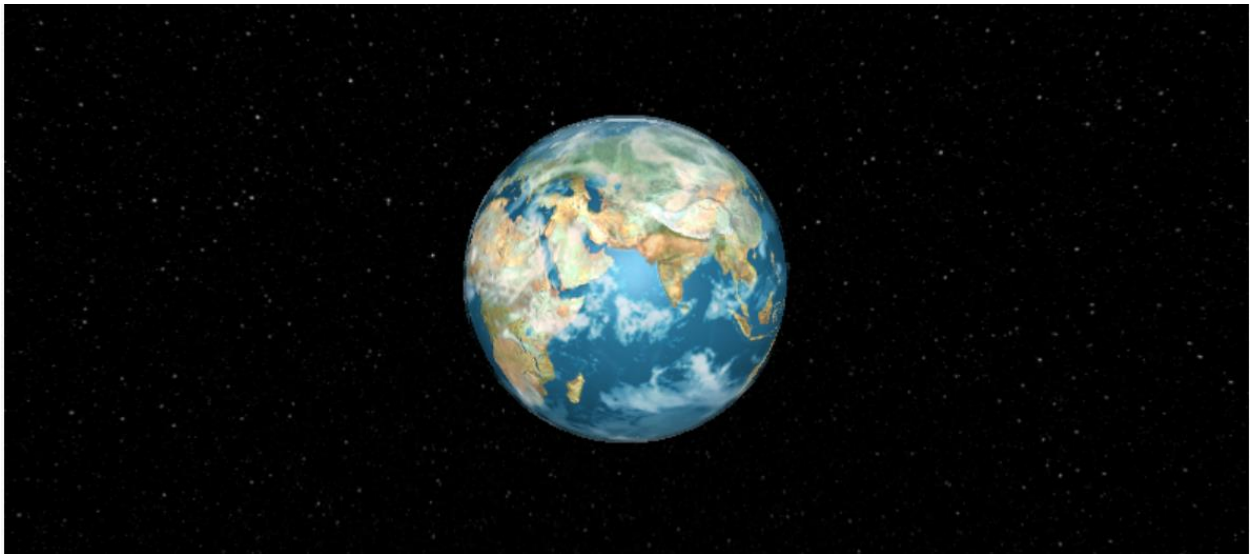
В качестве траекторий движения планет и Луны можно использовать окружность.

Расстояния между планетами, их скорость вращения вокруг собственной оси и солнца должны отражать реальные отношения размеров, расстояний и скоростей. (Марс меньше Земли, Меркурий меньше Марса и т.д.)

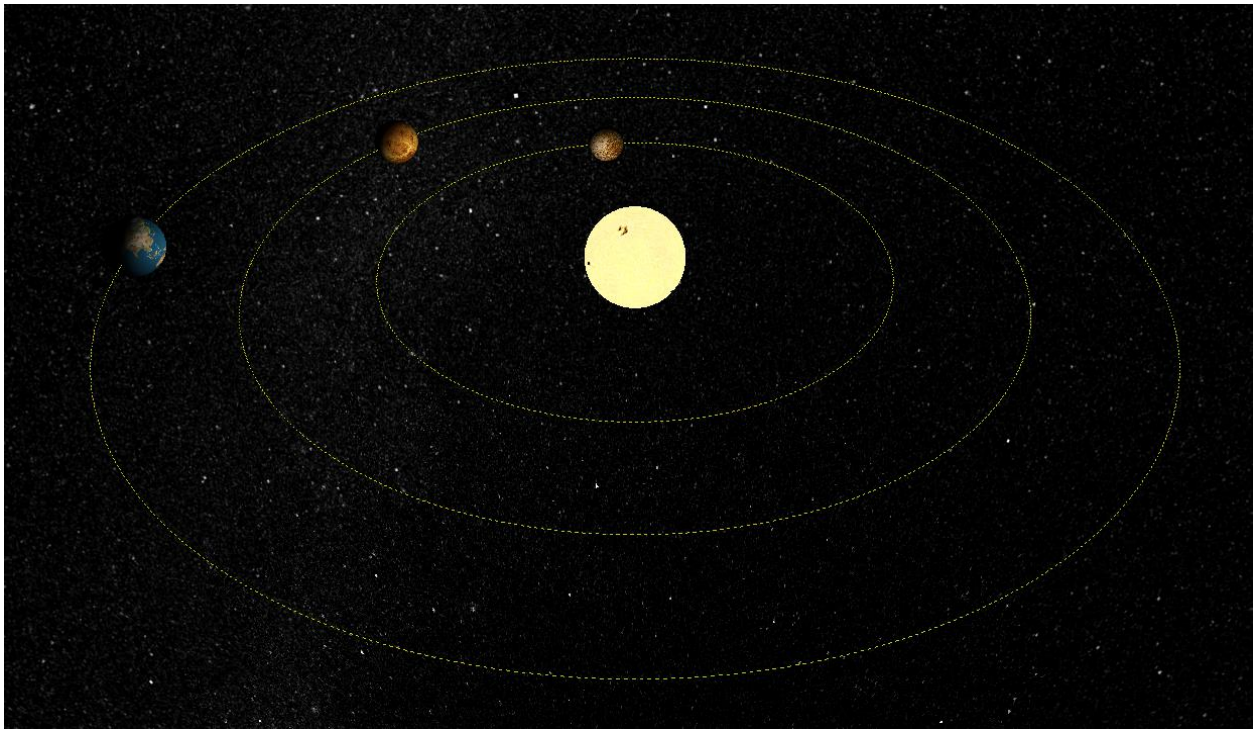
Также требуется реализовать режим слежения за планетами. По нажатию на клавиши 1 – 4 фокус камеры должен смещаться вслед за позицией планеты,

соответствующей номеру нажатой клавиши. По нажатию кнопки 0 должен включаться общий вид на Солнечную систему.

Режим слежения:



Общий вид:



Листинг программы:

```
// ссылка на блок веб-страницы, в котором будет отображаться графика
var container;
```

```
// переменные: камера, сцена, отрисовщик, наблюдатель
var camera, scene, renderer, choice = 0;

// создание загрузчика текстур
var loader = new THREE.TextureLoader();

// массив планет
var planets = [];

// нажатие клавиш
var keyboard = new THREEEx.KeyboardState();

// время
var clock = new THREE.Clock();

// в этой функции можно добавлять объекты и выполнять их первичную настройку
function init()
{
    // получение ссылки на блок html-страницы
    container = document.getElementById('container');
    // создание сцены
    scene = new THREE.Scene();

    // установка параметров камеры
    // 45 - угол обзора
    // window.innerWidth / window.innerHeight - соотношение сторон
    // 1 и 4000 - ближняя и дальняя плоскости отсечения
    camera = new THREE.PerspectiveCamera(45, window.innerWidth /
window.innerHeight, 1, 4000);

    // установка позиции камеры
    camera.position.set(0, 40, 35);

    // установка точки, на которую камера будет смотреть
    camera.lookAt(new THREE.Vector3(0, 0, 0));

    // создание отрисовщика
    renderer = new THREE.WebGLRenderer( { antialias: false } );
    renderer.setSize(window.innerWidth, window.innerHeight);

    // закрашивание экрана синим цветом, заданным в шестнадцатеричной системе
    renderer.setClearColor(new THREE.Color(0.5,0.5,0.5), 1);

    container.appendChild(renderer.domElement);

    // добавление обработчика события изменения размеров окна
    window.addEventListener('resize', onWindowResize, false);

    // звездное небо
    createStars();
```

```
    // планеты
    createPlanets();

    // свет
    createLight();
}

function onWindowResize()
{
    // изменение соотношения сторон для виртуальной камеры
    camera.aspect = window.innerWidth / window.innerHeight;
    camera.updateProjectionMatrix();

    // изменение соотношения сторон рендера
    renderer.setSize(window.innerWidth, window.innerHeight);
}

// в этой функции можно изменять параметры объектов и обрабатывать действия
пользователя
function animate()
{
    // сколько прошло времени
    let deltaTime = clock.getDelta()

    // изменяем положение планеты
    for(let i = 0; i < planets.length - 1; i++) {

        // изменение угла
        planets[i].angle += planets[i].speed * deltaTime;
        if(planets[i].angle >= 360 || planets[i].angle <= -360){
            planets[i].angle = 0;
        }

        // изменение координат
        let r = planets[0].pos.distanceTo(planets[i].pos)
        planets[i].pos.x = r * Math.cos(planets[i].angle * 3.14 / 180)
        planets[i].pos.z = r * Math.sin(planets[i].angle * 3.14 / 180)
        planets[i].sphere.position.copy(planets[i].pos)
        planets[i].sphere.rotation.y = planets[i].angle * 3.14 / 180
    }

    // просчет позиции и угла для луны
    // угла
    let j = planets.length - 1
    planets[j].angle += planets[j].speed * deltaTime;
    if(planets[j].angle >= 360 || planets[j].angle <= -360){
        planets[j].angle = 0;
    }

    // координат
```

```
    planets[j].pos.x = planets[3].pos.x + 2 * Math.cos(planets[j].angle * 3.14 /
180)
    planets[j].pos.z = planets[3].pos.z + 2 * Math.sin(planets[j].angle * 3.14 /
180)
    planets[j].sphere.position.copy(planets[j].pos)
    planets[j].sphere.rotation.y = planets[j].angle * 3.14 / 180

    // режим слежения
    if (keyboard.pressed("0")) {

        // общий вид
        choice = 0
    }
    if(keyboard.pressed("1")) {

        // меркурий
        choice = 1
    }
    if(keyboard.pressed("2")) {

        // венера
        choice = 2
    }
    if(keyboard.pressed("3")) {

        // земля
        choice = 3
    }
    if(keyboard.pressed("4")) {

        // марс
        choice = 4
    }

    // смена позиции камеры
    if(choice == 0) {
        camera.position.set(0, 40, 35);
        camera.lookAt(new THREE.Vector3(0, 0, 0));
    }
    else {
        camera.position.set(planets[choice].pos.x - 6, 5, planets[choice].pos.z -
6);
        camera.lookAt(planets[choice].pos);
    }

    // добавление функции на вызов при перерисовке браузером страницы
    requestAnimationFrame(animate);
    render();
}

function render()
```

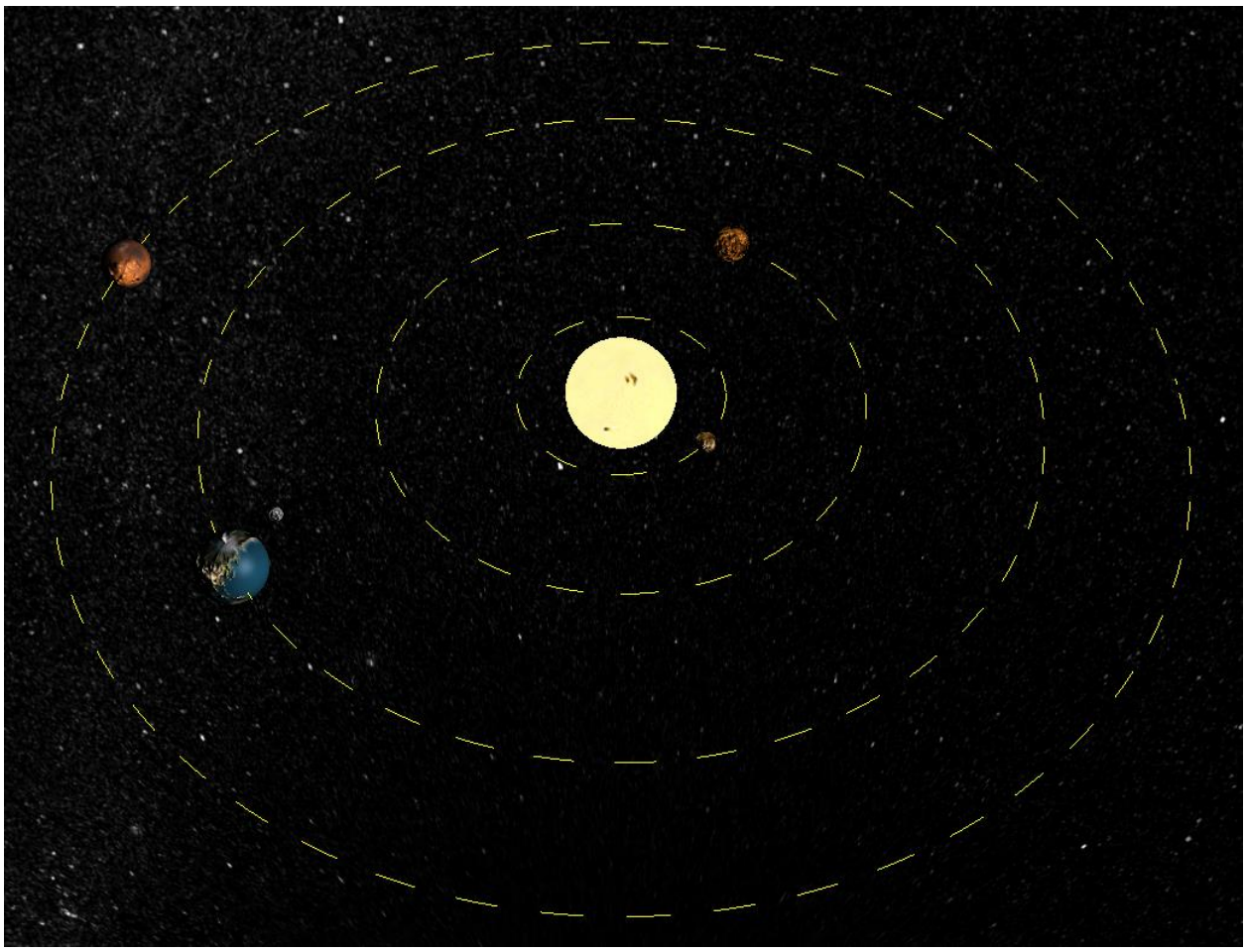
```
{
    // рисование кадра
    renderer.render(scene, camera);
}

// создание планеты
function createPlanet(tex_path, bump_path, r, x0, speed) {
    let planet = {};
    let geometry = new THREE.SphereGeometry(r, 32, 32);
    let tex = loader.load(tex_path);
    let material = null;
    if(bump_path != "") {
        let bump = loader.load(bump_path);
        material = new THREE.MeshPhongMaterial({
            map: tex,
            bumpMap: bump,
            bumpScale: 0.5,
            side: THREE.DoubleSide
        });
    }
    else {
        material = new THREE.MeshBasicMaterial({
            map: tex,
            side: THREE.DoubleSide
        });
    }
    planet.sphere = new THREE.Mesh(geometry, material);
    planet.pos = new THREE.Vector3(x0, 0, 0);
    planet.angle = 0
    planet.speed = speed
    return planet;
}

// отрисовка линии объекта
function drawLine(r) {

    // создаем примитив
    var lineGeometry = new THREE.Geometry();
    var vertices = lineGeometry.vertices;

    // отмечаем сегменты
    for(let i = 0; i < 360; i++) {
        let x = r * Math.cos(i * 3.14 / 180)
        let z = r * Math.sin(i * 3.14 / 180)
        vertices.push(new THREE.Vector3(x, 0, z));
    }

    // параметры: цвет, размер черты, размер промежутка
    var lineMaterial = new THREE.LineDashedMaterial({
        color: new THREE.Color(1, 1, 0),
        dashSize: 1,
```

```
        gapSize: 1
    });
    var line = new THREE.Line(lineGeometry, lineMaterial);
    line.computeLineDistances();
    scene.add(line);
}

// звездное небо
function createStars() {

    // создание геометрии для сферы
    var geometry = new THREE.SphereGeometry(64, 32, 32);

    // загрузка текстуры
    var tex = loader.load("planets/starmap.jpg");

    // создание материала
    var material = new THREE.MeshBasicMaterial({
        map: tex,
        side: THREE.DoubleSide
    });

    // создание объекта
    var sphere = new THREE.Mesh(geometry, material);

    // размещение объекта в сцене
    scene.add(sphere);
}

// начальное формирование планет
function createPlanets() {

    // солнце
    var sun = createPlanet("planets/sunmap.jpg", "", 1.6, 0, 10)
    planets.push(sun);

    // меркурий
    var mercury = createPlanet("planets/mercury/mercurymap.jpg",
"planets/mercury/mercurybump.jpg", 0.3, 3, 80)
    drawLine(sun.pos.distanceTo(mercury.pos))
    planets.push(mercury);

    // венера
    var venus = createPlanet("planets/venus/venusmap.jpg",
"planets/venus/venusbump.jpg", 0.58, 7, 60)
    drawLine(sun.pos.distanceTo(venus.pos))
    planets.push(venus)

    // земля
    var earth = createPlanet("planets/earth/earthmap1k.jpg",
"planets/earth/earthbump1k.jpg", 1, 12, 30)
```

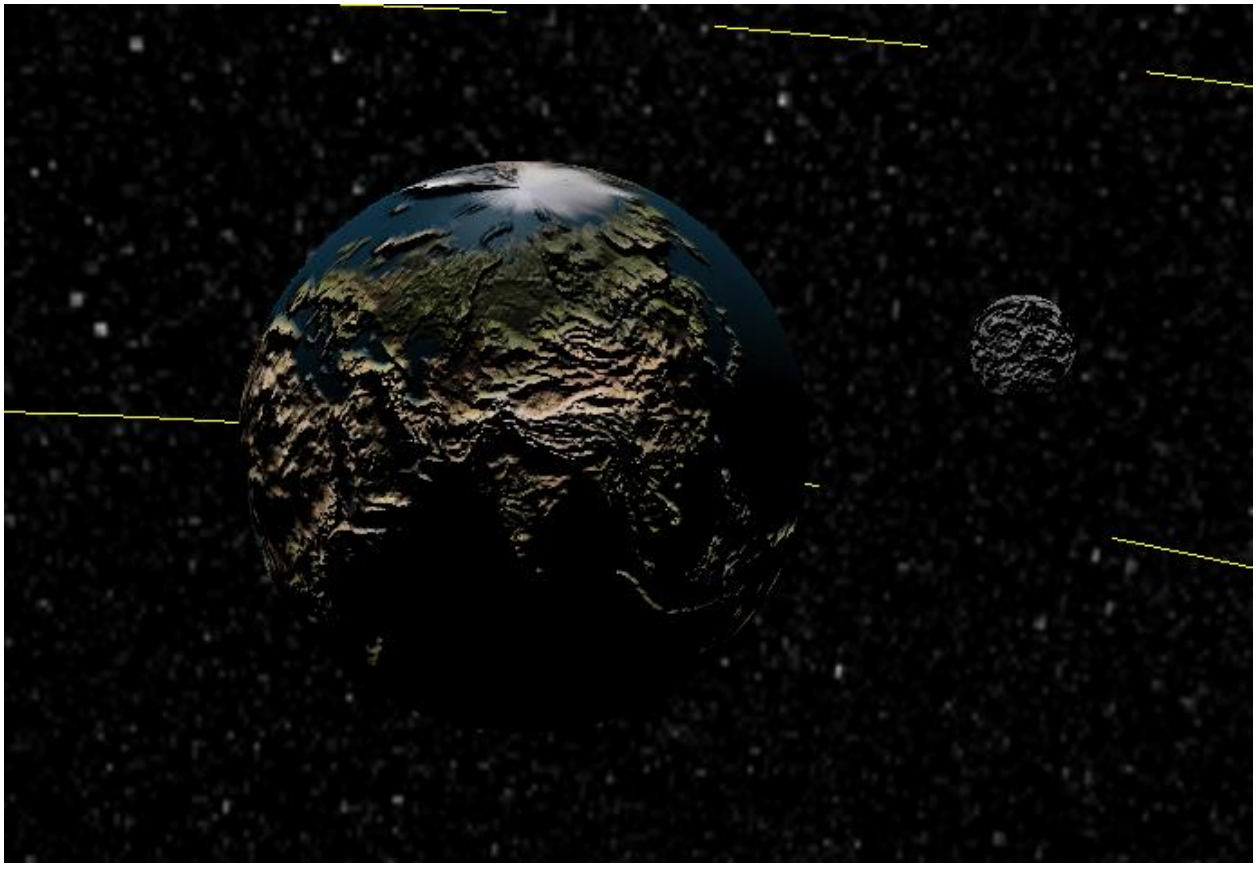
```
    drawLine(sun.pos.distanceTo(earth.pos))
    planets.push(earth)

    //марс
    var mars = createPlanet("planets/mars/marsmap1k.jpg",
"planets/mars/marsbump1k.jpg", 0.75, 16, 40)
    drawLine(sun.pos.distanceTo(mars.pos))
    planets.push(mars)

    // луна - спутник земли
    var moon = createPlanet("planets/earth/moon/moonmap1k.jpg",
"planets/earth/moon/moonbump1k.jpg", 0.2, 14, 60)
    planets.push(moon)

    // перебор планет
    for (var i = 0; i < planets.length; i++)
    {
        // действие
        planets[i].sphere.position.copy(planets[i].pos)
        scene.add(planets[i].sphere)
    }
}

// свет
function createLight() {

    // создание точечного источника освещения заданного цвета
    var spotlight = new THREE.PointLight(new THREE.Color(1, 1, 1));

    // установка позиции источника освещения
    spotlight.position.set(20, 20, 20);

    // добавление источника в сцену
    scene.add(spotlight);
}

// функция инициализации камеры, отрисовщика, объектов сцены и т.д.
init();

// обновление данных по таймеру браузера
animate();
```

Результат работы программы:

Общий вид:



Режим слежения:



Вывод:

В ходе выполнения лабораторной работы были изучены способы перемещения объектов в трёхмерном пространстве посредством графической библиотеки Three.js.

Ссылка на GitHub:

https://github.com/russkih1984/LR2-OTG