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 № 8

по дисциплине "Программирование графических приложений"

ТЕМА РАБОТЫ:

Матрицы и геометрические преобразования трехмерных объектов

Выполнил:

студент гр. ПРИм-124

Парахин К.В.

Принял:

Жигалов И.Е.

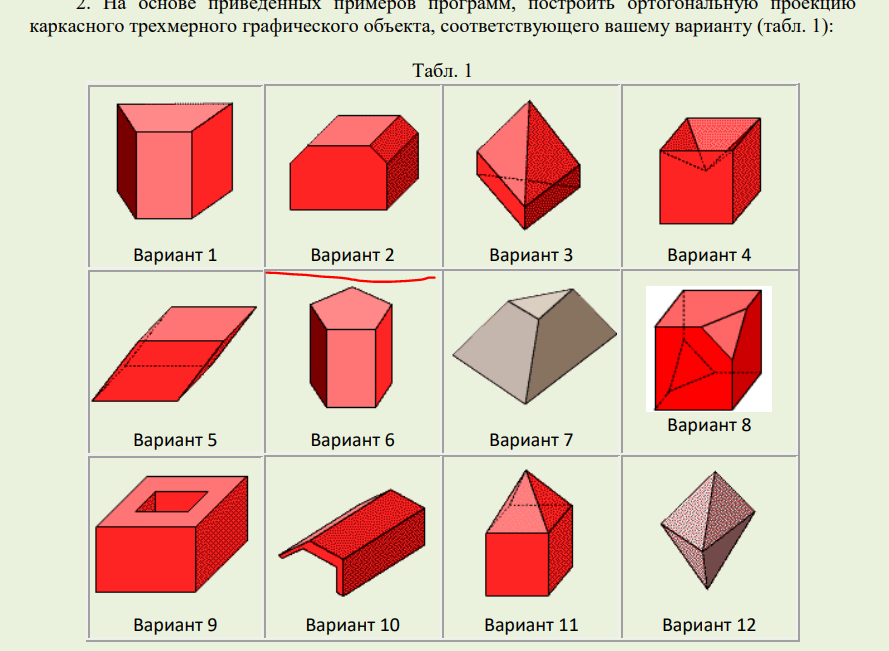
Владимир 2024 г.

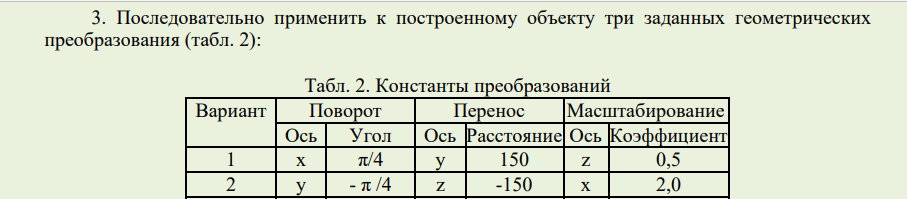
Цель работы:

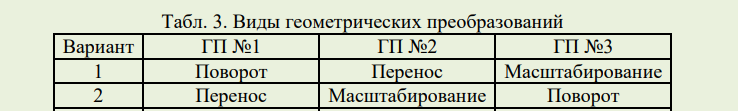
Изучение методов формирования трехмерных графических объектов и использования матричных операций в WebGL.

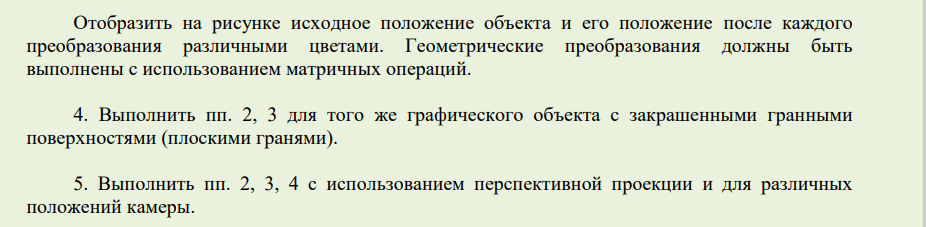
Выполнение работы:

Индивидуальный вариант 2 (14 % 12 = 2)









var scene, camera, renderer, box, new\_axis;

var dx=1, dy=0, dz=0;

var sx = 1, sy = 1, sz = 1;

var direct=1;

var delta = -Math.PI/4;

var angle=0;

var positionConstX;

var positionConstY;

var positionConstZ;

init();

animate();

function init(){

    renderer = new THREE.WebGLRenderer();

    renderer.setSize( 800, 600 );

    document.body.appendChild( renderer.domElement );

    camera = new THREE.PerspectiveCamera( 70, 800 / 600, 1, 1000 );

    camera.translateX( 50 );

    camera.translateY( 70 );

    camera.translateZ( 200 );

    scene = new THREE.Scene();

    var box\_material = new THREE.MeshLambertMaterial( { color: 0x00FF00 } );

    var width = 30;

    var height = 30;

    var depth = 30;

    var roundCornerWidth = 5;

    var roundCornerHeight = 7;

    var helpWidth = 4;

    var helpHeight = 5;

    var box\_geometry = new THREE.BoxGeometry( width, height, depth, 1, 1, 1 );

    box\_geometry.vertices.forEach(v => {

        if(Math.abs(v.x)>helpWidth/2){

          if(Math.abs(v.y)>helpHeight/2){

            let helperX = Math.abs(v.x)-helpWidth/2;

            let helperY2 = (Math.abs(v.y)-helpHeight/2)/roundCornerHeight;

            let helperY = (1-helperX/roundCornerWidth) \* roundCornerHeight \* helperY2;

            v.y = Math.sign(v.y)\*((helpHeight/2 + helperY)+(Math.sin(helperX/roundCornerWidth \* Math.PI)\*(roundCornerHeight/4))\*helperY2);

            v.x = Math.sign(v.x)\*(Math.abs(v.x)+(Math.sin(helperX/roundCornerWidth \* Math.PI)\*(roundCornerWidth/4))\*helperY2);

          }

        }

      });

    box = new THREE.Mesh( box\_geometry, box\_material );

    var light1 = new THREE.AmbientLight( 0x666666 );

    var light2 = new THREE.PointLight( 0x666666, 1, 400 );

    light2.position.set( 200, 200, 50 );

    new\_axis = new THREE.Object3D();

    new\_axis.position.set( 50, 50, 0 );

    new\_axis.rotateOnAxis( new THREE.Vector3( 0, 1, 1 ).normalize(), 0.5 );

    var axisHelper\_scene = new THREE.AxisHelper( 100 );

    var axisHelper\_mesh = new THREE.AxisHelper( 50 );

    new\_axis.add( axisHelper\_mesh, box );

    scene.add( axisHelper\_scene, light1, light2, new\_axis );

}

var i = 0;

function animate(){

    // Перемещение

    if (i < 100){

        translating();

        i += 1;

    }

    if (i == 100){

        positionConstX = box.position.x;

        positionConstY = box.position.y;

        positionConstZ = box.position.z;

    }

    // Масштабирование

    if ((i >= 100) && (i < 115)){

        scaling();

        renderer.render( scene, camera );

        requestAnimationFrame( animate );

        i += 1;

    }

    if (i == 115){

        box.geometry.scale(1, 1, 1);

    }

    // Поворот

    if ((i >= 101) && (i < 110)){

        rotating();

        renderer.render( scene, camera );

        requestAnimationFrame( animate );

        i += 1;

    }

    renderer.render( scene, camera );

    requestAnimationFrame( animate );

}

function translating(){

    box.translateOnAxis( new THREE.Vector3( 1, 0, 0 ), dx\*direct );

    box.translateY( dy\*direct );

    box.translateOnAxis( new THREE.Vector3( 0, 0, 13 ).normalize(), dz\*direct );

    if ( box.position.x>100 ) {

        direct=-1;

    }

    if ( box.position.y>100 ) {

        direct=-1;

    }

    if ( box.position.z>100 ) {

        direct=-1;

    }

    if ( box.position.x<0 ){

        box.position.x=0;

        direct=1;

        dx=0;

        dy=1

    }

    if ( box.position.y<0 ){

        box.position.y=0;

        direct=1;

        dy=0;

        dz=1

    }

    if ( box.position.z<0 ){

        box.position.z=0;

        direct=1;

        dz=0;

        dx=1

    }

}

function scaling(){

    sx = 1 + (i - 100) / 100;

    if (sx == 2){

        return;

    }

    box.geometry.scale(sx,sy,sz);

/\*  box.position.x = positionConstX \* sx;

    box.position.y = positionConstY \* sy;

    box.position.z = positionConstZ \* sz; \*/

}

function rotating(){

    box.rotateOnAxis(new THREE.Vector3(1,0,0),dx);

    box.rotateY(dy);

    box.rotateOnAxis(new THREE.Vector3(0,0,99).normalize(),dz);

    angle+=delta;

    if(angle>Math.PI/2){

        angle=0;

        if(dx>0){

            dx=0;

            dy=delta;

        }

        else if(dy>0){

            dy=0;

            dz=delta;

        }

        else if(dz>0){

            dz=0;

            dx=delta;

        }

    }

}

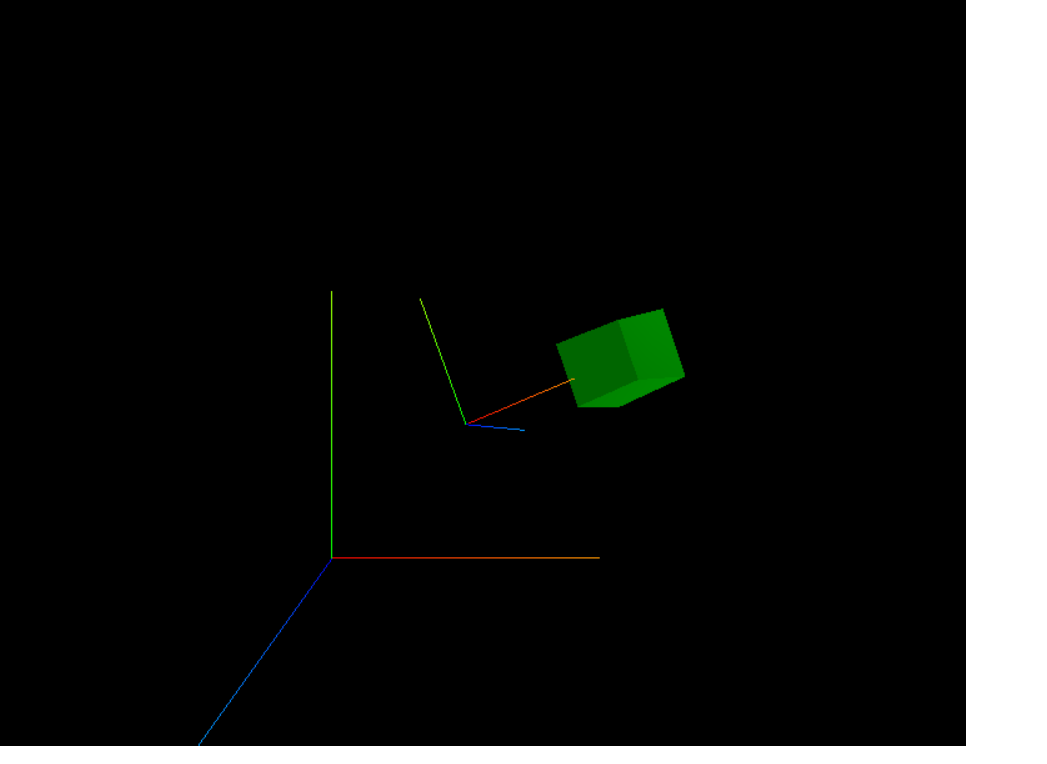


Рисунок 1. Перемещенный объект

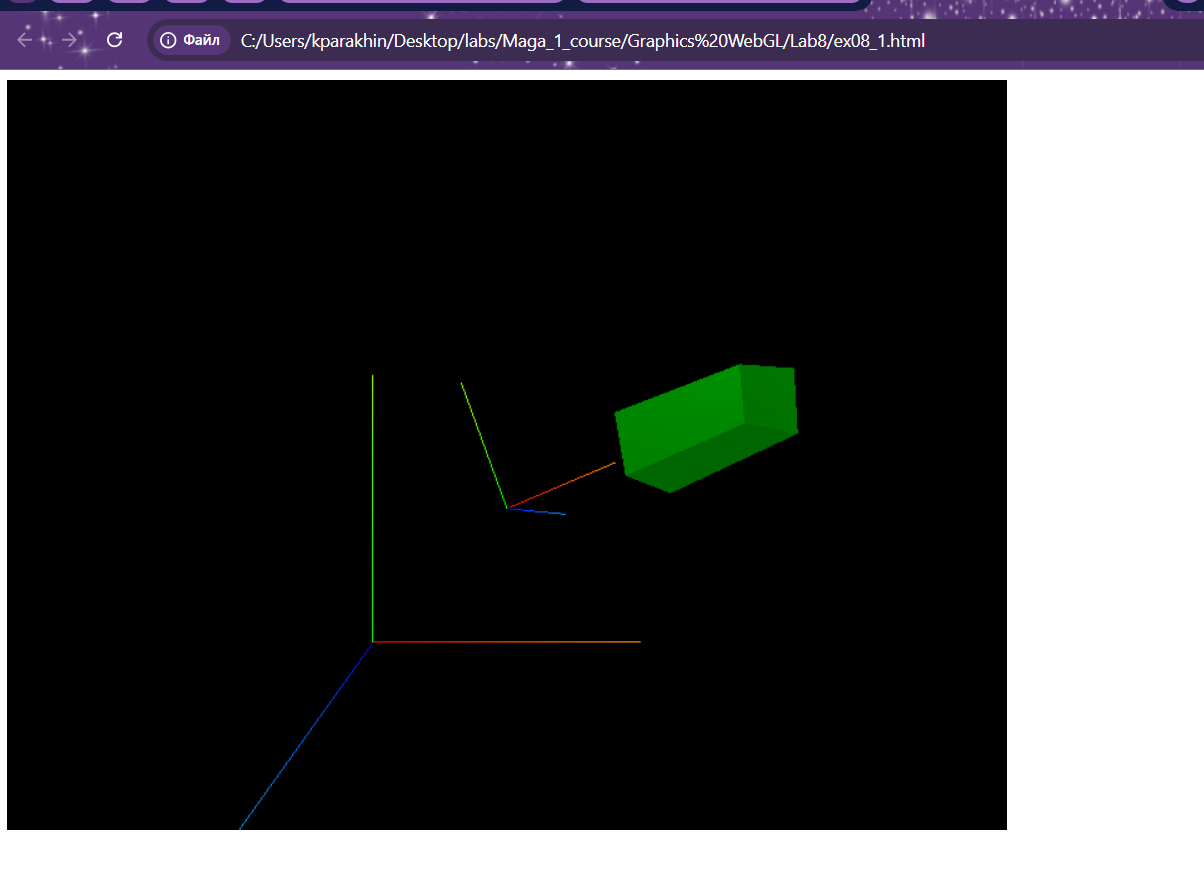


Рисунок 2. Масштабированный по OX и повернутый объект

Тут я сделал только геометрию обычного плоского объекта с плоскими гранями и базовой заливкой, его анимацию перемещения, масштабирования и поворота. Остальные действия вызвали некоторые сложности.

Вывод

В результате выполнения работы я провел изучение методов формирования трехмерных графических объектов и использования матричных операций в WebGL.