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 №1

по дисциплине "Программирование графических приложений"

ТЕМА РАБОТЫ:

Введение в WebGL

Выполнил:

студент гр. ПРИм-124

Парахин К.В.

Принял:

Жигалов И.Е.

Владимир 2024 г.

Цель работы:

Освоение принципов программирования веб-приложений с использованием WebGL.

Выполнение работы:

Индивидуальный вариант 2 (14 % 12 = 2)

1. Необходимо построить программу для вывода следующего плоского объекта – без использования библиотек



Листинг:

function initBuffers() {

 // ��������� ������ ������

  vertexBuffer = gl.createBuffer();

  gl.bindBuffer(gl.ARRAY\_BUFFER, vertexBuffer);

  // ������ ��������� ������ �������

  // ��������� ���-�� �����

  vertexBuffer.itemSize = 3;

  vertexBuffer.numberOfItems = 3;

}

function draw() {

    // ��������� ������� ���������

    // ���������, ��� ������ ������� ����� �� ��� ���������� (x, y, z)

    gl.vertexAttribPointer(shaderProgram.vertexPositionAttribute,

                         vertexBuffer.itemSize, gl.FLOAT, false, 0, 0);

    // ��������� ���������� - �������������

    gl.drawArrays(gl.TRIANGLES, 0, vertexBuffer.numberOfItems);

}

window.onload=function(){

    // �������� ������� canvas

    var canvas = document.getElementById("canvas3D");

    try {

        // ������� �������� �������� ����������� �������� WebGL.

        // ���� �� ���������, ���������� � ������������������ ���������

        gl = canvas.getContext("webgl") || canvas.getContext("experimental-webgl");

    }

    catch(e) {}

    // ���� �������� �� ������� ��������, ������� ���������

      if (!gl) {

        alert("Hello, WebGL");

      }

    if(gl){

        // ��������� �������� ������� ���������

        gl.viewportWidth = canvas.width;

        gl.viewportHeight = canvas.height;

        // ��������� ��������

        initShaders();

        // ��������� ������ ����� �������

        initBuffers();

        // �������� ���

        gl.clearColor(1.0, 0.0, 1.0, 0.5);

        // ��������� �����

        gl.viewport(0, 0, gl.viewportWidth, gl.viewportHeight);

        gl.clear(gl.COLOR\_BUFFER\_BIT);

        var triangle1Vertices = [

         0.4, -0.1,  0.0,

         0.0,  0.0,  0.0,

         0.4,  0.1,  0.0

        ];

            gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(triangle1Vertices), gl.STATIC\_DRAW);

        draw();

        var triangle2Vertices = [

         0.4,  0.1,  0.0,

         0.5,  0.5,  0.0,

         0.6,  0.1,  0.0

        ];

            gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(triangle2Vertices), gl.STATIC\_DRAW);

        draw();

        var triangle3Vertices = [

         0.6,  0.1,  0.0,

         1.0,  0.0,  0.0,

         0.6,  -0.1,  0.0

        ];

            gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(triangle3Vertices), gl.STATIC\_DRAW);

        draw();

        var triangle4Vertices = [

         0.6,  -0.1,  0.0,

         0.5,  -0.5,  0.0,

         0.4,  -0.1,  0.0

        ];

            gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(triangle4Vertices), gl.STATIC\_DRAW);

        draw();

        var triangle5Vertices = [

         0.4,  -0.1, 0.0,

         0.4,  0.1,  0.0,

         0.6,  0.1,  0.0

        ];

            gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(triangle5Vertices), gl.STATIC\_DRAW);

        draw();

        var triangle6Vertices = [

         0.4,  -0.1, 0.0,

         0.6,  0.1,  0.0,

         0.6,  -0.1,  0.0,

        ];

            gl.bufferData(gl.ARRAY\_BUFFER, new Float32Array(triangle6Vertices), gl.STATIC\_DRAW);

        draw();

    }

}

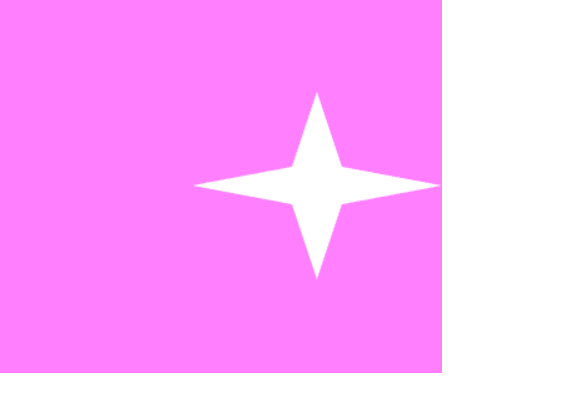
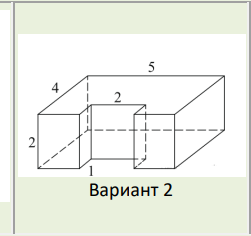


Рисунок 1. Выполнение задания 1

1. Построить структурированную программу для вывода объемного графического объекта



Для начала установим библиотеку three.min.js и TrackballControls.js

Используем куб из примера, меняем его ориентацию и убираем анимацию вращения. Затем в нужном месте делаем визуальный вырез – то есть куб со цветом сцены (белый) – и получаем следующий результат

var geometry = new THREE.BoxGeometry( 200, 100, 150);

var material = new THREE.MeshBasicMaterial( { color: 0x00ff00 } );

Cube = new THREE.Mesh( geometry, material );

Cube.position.z = -100;

Cube.rotation.y = -Math.PI / 12;

scene.add( Cube );

var geometry = new THREE.BoxGeometry( 100, 100, 30);

var material = new THREE.MeshBasicMaterial( { color: 0xffffff } );

Cube = new THREE.Mesh( geometry, material );

Cube.position.z = 0;

Cube.position.y = 20;

Cube.rotation.y = -Math.PI / 12;

scene.add( Cube );

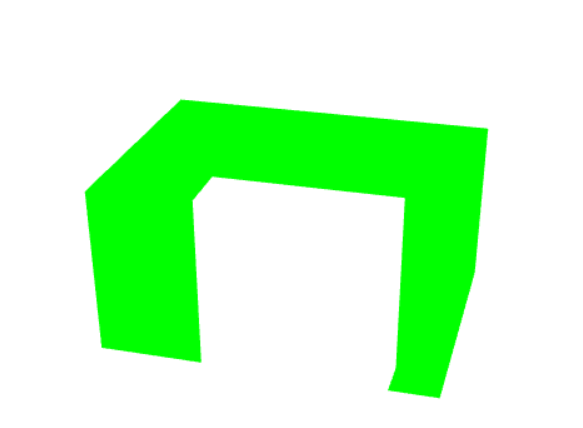


Рисунок 2. Выполнение задания 2

Вывод

В результате выполнения работы я освоил основные принципы программирования графических веб-приложений с использованием WebGL.