МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 5

з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”

Виконав
студент ІІІ курсу
групи КП-82
Кривчук Денис
(*прізвище, ім’я, по батькові*)

Зарахована
“____” “_____” 20___ р.
викладачем
Шкурат Оксаною Сергіївною
(*прізвище, ім’я, по батькові*)

варіант № 9

Київ 2020

Варіант завдання

Завдання: Імпортувати моделі тривимірних об’єктів форматів, що визначені варіантом. Створити реалістичну анімацію об’єкту. Додати до сцени фон, інші об’єкти для надання сцені реалістичного вигляду. Для цього використати текстури, матеріали, імпортувати додаткові об’єкти з відкритих бібліотек, за бажанням створити прості об’єкти у графічному редакторі. Студенти, які мають непарний номер варіанту у списку групи імпортують моделі формату .obj, парний варіант – .lwo.

Варіант: 9

Лістинг коду програми

Myanimation.java

```
package pack;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import javax.media.j3d.*;
import javax.swing.JFrame;
import javax.swing.Timer;
import javax.vecmath.*;

public class MyAnimation  implements ActionListener, KeyListener{

    private TransformGroup wholePlane;
    private Transform3D translateTransform;
    private Transform3D rotateTransformX;
    private Transform3D rotateTransformY;
    private Transform3D rotateTransformZ;
    private Transform3D scaleTransform;


    private JFrame mainFrame;

    private float rot_angle = 0.f;
    private float sign=1.0f;
    private float zoom=0.5f;
    private float xloc=0.3f;
    private float yloc=0.3f;
    private float zloc=0.0f;
    private int moveType=1;
    private Timer timer;

    public MyAnimation(TransformGroup wholePlane, Transform3D trans,JFrame
frame){
        this.wholePlane=wholePlane;
        this.translateTransform=trans;
        this.mainFrame=frame;

        rotateTransformX= new Transform3D();
        rotateTransformY= new Transform3D();
        rotateTransformZ= new Transform3D();

        timer = new Timer(100, this);

        timer.start();
    }

    private void initialPlaneState(){
        xloc=0.0f;
        yloc=0.0f;
        zloc=0.0f;
        zoom=0.2f;
        moveType=1;
        sign=1.0f;
        rotateTransformY.rotY(-Math.PI/2.8);
        translateTransform.mul(rotateTransformY);
        if(timer.isRunning()){timer.stop();}
    }
```

```
    @Override
    public void actionPerformed(ActionEvent e) {
        // start timer when button is pressed

        Move(moveType);
        translateTransform.setScale(new Vector3d(zoom,zoom,zoom));
        translateTransform.setRotation(new AxisAngle4d(0,yloc,0,rot_angle ));
        translateTransform.setTranslation(new Vector3f(xloc,yloc,zloc));
        wholePlane.setTransform(translateTransform);

    }

    private void Move(int mType) {
        xloc = (float) Math.sin(rot_angle);
        yloc = 0.6f *  (float) (Math.cos(rot_angle)-1);
        zoom = ((float) (-Math.cos(rot_angle)+1))/4f+0.5f;
        rot_angle += 0.1;
    }

    @Override
    public void keyTyped(KeyEvent e) {
        //Invoked when a key has been typed.
    }

    @Override
    public void keyPressed(KeyEvent e) {
        //Invoked when a key has been pressed.
    }

    @Override
    public void keyReleased(KeyEvent e) {
        // Invoked when a key has been released.
    }
}
```

MyScene.java

```
package pack;

//import com.microcrowd.loader.java3d.max3ds.Loader3DS;

import com.sun.j3d.utils.geometry.*;
import com.sun.j3d.utils.universe.*;

import java.awt.Color;
import javax.media.j3d.*;
import javax.media.j3d.Material;
import javax.vecmath.*;
import javax.media.j3d.Background;

import com.sun.j3d.loaders.*;
import com.sun.j3d.loaders.objectfile.ObjectFile;
import com.sun.j3d.utils.image.TextureLoader;
import java.awt.*;
import java.io.FileReader;
import java.io.IOException;
import java.util.Map;
import javax.swing.JFrame;

public class MyScene extends JFrame {
    static SimpleUniverse universe;
    static Scene scene;
    static Map<String, Shape3D> nameMap;
    static BranchGroup root;
    static Canvas3D canvas;
```

```
    static TransformGroup wholeModel;
    static Transform3D transform3D;

    public MyScene() throws IOException{
        configureWindow();
        configureCanvas();
        configureUniverse();
        addModelToUniverse();
        setModelElementsList();
        addAppearance();
        addImageBackground();
        addLightToUniverse();
        root.compile();
        universe.addBranchGraph(root);
//        addOtherLight();
        ChangeViewAngle();
    }
    private void configureWindow()  {
        setTitle("Animation Example");
        setSize(760,640);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }

    private void configureCanvas(){
        canvas=new Canvas3D(SimpleUniverse.getPreferredConfiguration());
        canvas.setDoubleBufferEnable(true);
        getContentPane().add(canvas,BorderLayout.CENTER);
    }

    private void configureUniverse(){
        root= new BranchGroup();
        universe= new SimpleUniverse(canvas);
        universe.getViewingPlatform().setNominalViewingTransform();
    }
    private void addModelToUniverse() throws IOException{
        scene = getSceneFromFile("d:\\objects\\toyplane.obj");
        root = scene.getSceneGroup();
    }

    private void printModelElementsList(Map<String,Shape3D> nameMap){
        for (String name : nameMap.keySet()) {
            System.out.printf("Name: %s\n", name);}
    }

    private void setModelElementsList() {
        nameMap=scene.getNamedObjects();
        //Print elements of your model:
        printModelElementsList(nameMap);

        wholeModel = new TransformGroup();
        transform3D = new Transform3D();
        transform3D.setScale(new Vector3d(0.5,0.5,0.5));
        wholeModel.setTransform(transform3D);
        root.removeChild(nameMap.get("plane"));
        wholeModel.addChild(nameMap.get("plane"));
        wholeModel.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        root.addChild(wholeModel);
    }
    Texture getTexture(String path) {
        TextureLoader textureLoader = new
TextureLoader(path,"LUMINANCE",canvas);
        Texture texture = textureLoader.getTexture();
        texture.setBoundaryModeS(Texture.WRAP);
```

```
        texture.setBoundaryModeT(Texture.WRAP);
        texture.setBoundaryColor( new Color4f( 0.0f, 1.0f, 0.0f, 0.0f ) );
        return texture;
    }

    Material getMaterial() {
        Material material = new Material();
        material.setAmbientColor ( new Color3f( 3.f, 3.f, 3.f ) );
        material.setDiffuseColor ( new Color3f( 5f, 0.11f, 3.f ) );
        material.setSpecularColor( new Color3f( 0.95f, 3.f, 3.f ) );
        material.setShininess( 0.3f );
        material.setLightingEnable(true);
        return material;
    }

    private void addAppearance(){
        Appearance planeAppearance = new Appearance();
        planeAppearance.setTexture(getTexture("d:\\objects\\texture.jpg"));
        TextureAttributes texAttr = new TextureAttributes();
        texAttr.setTextureMode(TextureAttributes.COMBINE);
        planeAppearance.setTextureAttributes(texAttr);
        planeAppearance.setMaterial(getMaterial());
        Shape3D plane = nameMap.get("plane");
        plane.setAppearance(planeAppearance);
    }

    private void addImageBackground(){
        TextureLoader t = new TextureLoader("d:\\objects\\sky.jpg", canvas);
        Background background = new Background(t.getImage());
        background.setImageScaleMode(Background.SCALE_FIT_ALL);
        BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0,
0.0),100.0);
        background.setApplicationBounds(bounds);
        root.addChild(background);
    }
    private void addLightToUniverse(){
        Bounds bounds = new BoundingSphere();
        Color3f color = new Color3f(65/255f, 30/255f, 25/255f);
        Vector3f lightdirection = new Vector3f(-1f,-1f,-1f);
        DirectionalLight dirlight = new
DirectionalLight(color,lightdirection);
        dirlight.setInfluencingBounds(bounds);
        root.addChild(dirlight);
    }
    public static Scene getSceneFromFile(String location) throws IOException
{
        ObjectFile file = new ObjectFile(ObjectFile.RESIZE);
        file.setFlags (ObjectFile.RESIZE | ObjectFile.TRIANGULATE |
ObjectFile.STRIPIFY);
        return file.load(new FileReader(location));
    }

    private void ChangeViewAngle(){
        ViewingPlatform vp = universe.getViewingPlatform();
        TransformGroup vpGroup =
vp.getMultiTransformGroup().getTransformGroup(0);
        Transform3D vpTranslation = new Transform3D();
        Vector3f translationVector = new Vector3f(0.0F, -1.2F, 6F);
        vpTranslation.setTranslation(translationVector);
        vpGroup.setTransform(vpTranslation);
    }
    public static void main(String[]args){
        try {
            MyScene window = new MyScene();
```

```
            MyAnimation planeMovement = new MyAnimation(wholeModel,
transform3D, window);
            window.addKeyListener(planeMovement);
            window.setVisible(true);
        }
        catch (IOException ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

Результат