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 №6

з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”
тема: “Анімація тривимірних об’єктів”

Виконав
студент III курсу
групи КП-83
Ландо Максим Юрійович
варіант № 9

Зарахована
“____” “____________” 20___ р.
викладачем
Шкурат Оксаною Сергіївною

Київ 2021

Мета: Навчитися анімувати складні об’єкти тривимірної сцени.

Задання на лабораторну роботу

Виконати анімацію тривимірної сцени за варіантом.

Варіант:

Анімація білка Скрата (із мультфільму) scrat.obj. Горіх повинен рухатися по екрану, білка – за ним; білка повинна рухати руками або ногами, хвостом.

Лістинг коду програми

MyAnimation.java

```java
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 TransformGroup left_hand;
    private Transform3D left_trans;
    private TransformGroup right_hand;
    private Transform3D right_trans;
    private TransformGroup tail;
    private Transform3D tail_trans;

    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,
                       TransformGroup left_hand,Transform3D left_trans,
                       TransformGroup right_hand, Transform3D right_trans,
                       TransformGroup tail, Transform3D tail_trans,
                       JFrame frame){
        this.tail = tail;
        this.tail_trans = tail_trans;
        this.left_hand = left_hand;
        this.left_trans = left_trans;
        this.right_hand = right_hand;
        this.right_trans = right_trans;
        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();
    }
    @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);

        float newangle = rot_angle*3;
        while(newangle<-2*Math.PI){
            newangle += 2*Math.PI;
        }

        while(Math.abs(newangle)>Math.PI/3){
            if (newangle>Math.PI/6){
                newangle = (float) (Math.PI/6 - newangle);
            }
            if (newangle< - Math.PI/6){
                newangle = (float) (-Math.PI/6 - newangle);
            }
        }
        left_trans.rotX(newangle);
        left_hand.setTransform(left_trans);
        right_trans.rotX(-newangle);
        right_hand.setTransform(right_trans);
        tail_trans.rotY(newangle);
        tail.setTransform(tail_trans);
    }

    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;
        if (rot_angle>2*Math.PI){
            rot_angle = 0f;
        }
    }

    @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.
    }
}
```

Squirrel.java

```
import com.sun.j3d.loaders.Scene;
import com.sun.j3d.loaders.objectfile.ObjectFile;
import com.sun.j3d.utils.image.TextureLoader;
import com.sun.j3d.utils.universe.SimpleUniverse;
import com.sun.j3d.utils.universe.ViewingPlatform;

import javax.media.j3d.*;
import javax.swing.*;
import javax.vecmath.*;
import java.awt.*;
import java.io.FileReader;
import java.io.IOException;
import java.util.Map;

import javax.media.j3d.Material;

import javax.media.j3d.Background;
import javax.swing.JFrame;

public class Squirrel 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;
    static TransformGroup left_hand;
    static Transform3D left_trans;
    static TransformGroup right_hand;
    static Transform3D right_trans;
    static TransformGroup tail;
    static Transform3D tail_trans;

    public Squirrel() throws IOException {
        configureWindow();
        configureCanvas();
        configureUniverse();
        addModelToUniverse();
        setModelElementsList();
        addAppearance();
        addImageBackground();
        addLightToUniverse();
        root.compile();
        universe.addBranchGraph(root);
        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("c:\\objects\\scrat.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();
        printModelElementsList(nameMap);
        tail = new TransformGroup();
        wholeModel = new TransformGroup();
        left_hand = new TransformGroup();
        root.removeChild(nameMap.get("left_hand"));
        root.removeChild(nameMap.get("tale"));
        left_hand.addChild(nameMap.get("left_hand"));
        right_hand= new TransformGroup();
        root.removeChild(nameMap.get("right_hand"));
        right_hand.addChild(nameMap.get("right_hand"));
        transform3D = new Transform3D();
        left_trans = new Transform3D();
        tail_trans = new Transform3D();
        left_hand.setTransform(left_trans);
        right_trans = new Transform3D();
        right_hand.setTransform(right_trans);
        transform3D.setScale(new Vector3d(0.5,0.5,0.5));
        wholeModel.setTransform(transform3D);
        tail.addChild(nameMap.get("tale"));
        for (Map.Entry<String, Shape3D> entry : nameMap.entrySet()) {
            if (!entry.getKey().equals("left_hand") && !entry.getKey().equals("righ
t_hand")&& !entry.getKey().equals("tale")){
                root.removeChild(entry.getValue());
                wholeModel.addChild(entry.getValue());
            }
        }
        wholeModel.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        left_hand.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        right_hand.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        tail.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        root.addChild(wholeModel);
        wholeModel.addChild(left_hand);
        wholeModel.addChild(right_hand);
        wholeModel.addChild(tail);
    }
    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, 4f, 3.f ) );
        material.setSpecularColor( new Color3f( 5f, 3.f, 3.f ) );
        material.setSpecularColor( new Color3f( 5f, 3.f, 3.f ) );
        material.setShininess( 0.3f );
        material.setLightingEnable(true);
        return material;
    }

    Material getNutMaterial(){
        Material material = new Material();
        material.setAmbientColor ( new Color3f( 0.33f, 0.26f, 0.23f ) );
        material.setDiffuseColor ( new Color3f( 0.50f, 0.11f, 0.00f ) );
        material.setSpecularColor( new Color3f( 0.95f, 0.73f, 0.00f ) );
        material.setShininess( 0.3f );
        material.setLightingEnable(true);
        return material;
    }
    private void addAppearance(){
        Appearance bodyAppearance = new Appearance();
        Appearance nutAppearence = new Appearance();
        Texture t = getTexture("c:\\objects\\wood2.jpg");
        bodyAppearance.setTexture(t);
        nutAppearence.setTexture(t);
        TextureAttributes texAttr = new TextureAttributes();
        texAttr.setTextureMode(TextureAttributes.COMBINE);
        bodyAppearance.setTextureAttributes(texAttr);
        bodyAppearance.setMaterial(getMaterial());
        nutAppearence.setTextureAttributes(texAttr);
        nutAppearence.setMaterial(getNutMaterial());
        for (Map.Entry<String, Shape3D> entry : nameMap.entrySet()) {
            if (entry.getKey() == "nut"){
                entry.getValue().setAppearance(nutAppearence);
            }else{
                entry.getValue().setAppearance(bodyAppearance);
            }
        }

        Shape3D nut = nameMap.get("nut");
        nut.setAppearance(nutAppearence);

    }

    private void addImageBackground(){
        TextureLoader t = new TextureLoader("c:\\objects\\mountains.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.STRI
PIFY);
        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 {
            Squirrel window = new Squirrel();
            MyAnimation planeMovement = new MyAnimation(wholeModel, transform3D,
                    left_hand, left_trans, right_hand, right_trans,tail, tail_trans
, window);
            window.addKeyListener(planeMovement);
            window.setVisible(true);
        }
        catch (IOException ex) {
            System.out.println(ex.getMessage());
        }
    }
}
```

Результат