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 “Імпорт тривимірних моделей у середовище програмування java 3D, обробка та маніпуляція цих зображень.”

Виконав
студент III курсу
групи КП-82
Шило Андрій Володимирович
(прізвище, ім’я, по батькові)

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

варіант № 20

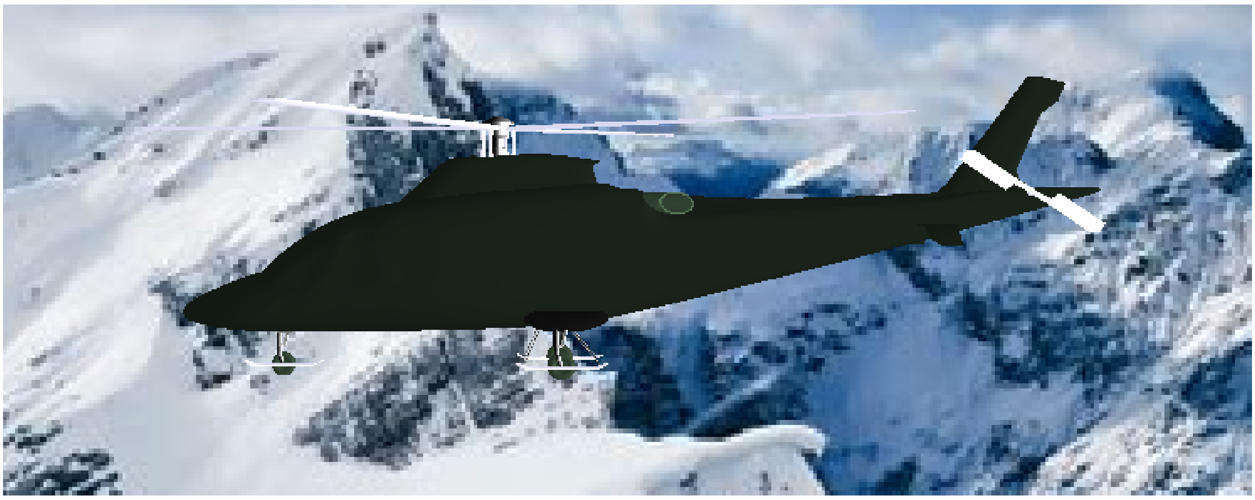
Київ 2021

Варіант завдання

Варіант: 20

гелікоптер

Результат:



Лістинг коду програми

```
package sample;
import javax.vecmath.*;
import com.sun.j3d.utils.universe.*;
import javax.media.j3d.*;
import com.sun.j3d.utils.behaviors.vp.*;
import com.sun.j3d.utils.image.TextureLoader;
import javax.swing.JFrame;
import com.sun.j3d.loaders.*;
import com.sun.j3d.loaders.objectfile.*;

import java.awt.*;
//
public class Main extends JFrame
{
        private final String helicopterPath = "helicopter.obj";
        private final String backgroundPath = "bg.jpg";
        public Canvas3D myCanvas3D;

        public Main()
        {
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        myCanvas3D = new Canvas3D(SimpleUniverse.getPreferredConfiguration());
        SimpleUniverse simpUniv = new SimpleUniverse(myCanvas3D);
        simpUniv.getViewingPlatform().setNominalViewingTransform();
        createSceneGraph(simpUniv);
        addLight(simpUniv);


        OrbitBehavior ob = new OrbitBehavior(myCanvas3D);
        ob.setSchedulingBounds(new BoundingSphere(new
Point3d(0.0,0.0,0.0),Double.MAX_VALUE));
        simpUniv.getViewingPlatform().setViewPlatformBehavior(ob);

        setTitle("Helicopter");
        setSize(948,604);
        getContentPane().add("Center", myCanvas3D);
        setVisible(true);
        }

        public void createSceneGraph(SimpleUniverse su)
        {
        BranchGroup theScene = new BranchGroup();
        Background background = new Background(new TextureLoader(backgroundPath,
myCanvas3D).getImage());
        background.setImageScaleMode(Background.SCALE_FIT_MAX);
        background.setApplicationBounds(new BoundingSphere(new Point3d(0, 0, 0),
Double.MAX_VALUE));
        background.setCapability(Background.ALLOW_IMAGE_WRITE);
        theScene.addChild(background);

        Scene helicopter = null;
        try
        {
                ObjectFile f = new ObjectFile(ObjectFile.RESIZE);
                f.setBasePath("D:/maokg/lab6");
                helicopter = f.load("helicopter.obj");
        }
        catch (Exception e)
        {
                System.out.println("File loading failed:" + e);
        }

        Transform3D scaling = new Transform3D();
        scaling.setScale(1.0/2);
        Transform3D helicopterTransform = new Transform3D();
        helicopterTransform.rotY(Math.PI*2);
        helicopterTransform.mul(scaling);
        TransformGroup helicopterTransformGroup = new
TransformGroup(helicopterTransform);
```

```
        TransformGroup sceneGroup = new TransformGroup();

        assert helicopter != null;
        BranchGroup helicopterSceneGroup = helicopter.getSceneGroup();
        helicopter.getNamedObjects().forEach((key, value) -> System.out.println(key + "
: " + value));

helicopterSceneGroup.removeChild((Shape3D)helicopter.getNamedObjects().get("cylinder.0
03_cylinder.004"));

helicopterSceneGroup.removeChild((Shape3D)helicopter.getNamedObjects().get("cylinder.0
04_cylinder.005"));

helicopterSceneGroup.removeChild((Shape3D)helicopter.getNamedObjects().get("cylinder.0
02"));

        sceneGroup.addChild(helicopter.getSceneGroup());

        sceneGroup.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        helicopterTransformGroup.addChild(sceneGroup);
        theScene.addChild(helicopterTransformGroup);

        Shape3D mainBody = (Shape3D) helicopter.getNamedObjects().get("cube");
        setAppearance(new Color(15, 20, 15), mainBody);

        Shape3D decal = (Shape3D) helicopter.getNamedObjects().get("cylinder");
        setAppearance(new Color(30, 40, 30), decal);

        Shape3D glass1 = (Shape3D)
helicopter.getNamedObjects().get("cube.006_cube.007");
        setAppearance(new Color(180, 180, 200), glass1);

        Shape3D glass2 = (Shape3D)
helicopter.getNamedObjects().get("cube.007_cube.008");
        setAppearance(new Color(180, 180, 200), glass2);

        Shape3D glass3 = (Shape3D)
helicopter.getNamedObjects().get("cylinder.004_cylinder.005");
        setAppearance(new Color(180, 180, 200), glass3);

        Shape3D smallPropeller = (Shape3D)
helicopter.getNamedObjects().get("cylinder.002");
        setAppearance(new Color(180, 180, 200), smallPropeller);

        Shape3D bigPropeller = (Shape3D)
helicopter.getNamedObjects().get("cylinder.003_cylinder.004");
        setAppearance(new Color(180, 180, 200), bigPropeller);

        Shape3D otherParts = (Shape3D)
helicopter.getNamedObjects().get("cube.001_cube.002");
        setAppearance(new Color(15, 20, 15), otherParts);

        Shape3D anotherParts = (Shape3D) helicopter.getNamedObjects().get("torus");
        setAppearance(new Color(30, 40, 30), anotherParts);

        Shape3D rocketHeadings = (Shape3D)
helicopter.getNamedObjects().get("cube.004_cube.005");
        setAppearance(new Color(10, 10, 10), rocketHeadings);

        Shape3D rockets = (Shape3D) helicopter.getNamedObjects().get("torus.001");
        setAppearance(new Color(30, 40, 30), rockets);

        Transform3D transformForBigPropeller = new Transform3D();
        transformForBigPropeller.setTranslation(new Vector3f(-0.22f, 0, 0));

        helicopterSceneGroup.addChild(applyRotationForShape(
                (Shape3D)helicopter.getNamedObjects().get("cylinder.003_cylinder.004"),
                transformForBigPropeller,
                1000
        ));
        helicopterSceneGroup.addChild(applyRotationForShape(
                (Shape3D)helicopter.getNamedObjects().get("cylinder.004_cylinder.005"),
                transformForBigPropeller,
                1000
        ));

        Transform3D transformForSmallPropeller = new Transform3D();
        transformForSmallPropeller.rotX(Math.PI/2);
        transformForSmallPropeller.setTranslation(new Vector3f(0.85f, 0.068f, 0));

        helicopterSceneGroup.addChild(applyRotationForShape(
                (Shape3D)helicopter.getNamedObjects().get("cylinder.002"),
                transformForSmallPropeller,
                500
        ));

        Transform3D transformMove = new Transform3D();
        transformMove.rotY(Math.PI);

        Alpha crawlAlpha = new Alpha(
                1, Alpha.INCREASING_ENABLE,0,0, 7000,0,0,0,0,0
        );
        PositionInterpolator positionInterpolator = new PositionInterpolator(
                crawlAlpha, sceneGroup, transformMove, -9.0f, 6.5f
        );

        BoundingSphere bs = new BoundingSphere(new Point3d(0,0,-600),
Double.MAX_VALUE);
        positionInterpolator.setSchedulingBounds(bs);
        sceneGroup.addChild(positionInterpolator);
        //com
        theScene.compile();
        su.addBranchGraph(theScene);
        }

        //com
        private void setAppearance(Color color, Shape3D shape) {
        Appearance app = new Appearance();
        Color3f color3f = new Color3f(color);
        app.setMaterial(new Material(color3f, color3f, color3f, color3f, 150.0f));
        shape.setAppearance(app);
        }
        //com
        private Node applyRotationForShape(Shape3D shape, Transform3D transform, int
rotateDuration) {
        TransformGroup transformGroup = new TransformGroup();
        transformGroup.addChild(shape.cloneTree());

        Alpha alpha = new Alpha(Integer.MAX_VALUE, Alpha.INCREASING_ENABLE,0,0,
rotateDuration,
                0,0,0, 0,0);
        RotationInterpolator rotationInterpolator = new RotationInterpolator(alpha,
transformGroup,
                transform, (float) Math.PI * 2, 0.0f);

        BoundingSphere bound = new BoundingSphere(new Point3d(), Double.MAX_VALUE);
        rotationInterpolator.setSchedulingBounds(bound);

        transformGroup.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        transformGroup.addChild(rotationInterpolator);

        return transformGroup;
        }

        public void addLight(SimpleUniverse su)
        {
        BranchGroup bgLight = new BranchGroup();
        BoundingSphere bounds = new BoundingSphere(new Point3d(0.0,0.0,0.0), 100.0);
        Color3f lightColour1 = new Color3f(1.0f,1.0f,1.0f);
        Vector3f lightDir1 = new Vector3f(-1.0f,0.0f,-0.5f);
        DirectionalLight light1 = new DirectionalLight(lightColour1, lightDir1);
        light1.setInfluencingBounds(bounds);

        bgLight.addChild(light1);
        su.addBranchGraph(bgLight);
        }


        public static void main(String[] args)
        {
        Main helicopter = new Main();
        }
}
```