

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”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконав  студент III курсу  групи КП-83  Клименко Ярослав  (*прізвище, ім’я, по батькові*)  варіант № 7 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2020

**Варіант завдання**

**Завдання**:

Анімація вертольоту helicopter.obj. У вертольота повинні рухатися обидва гвинти, вертоліт повинен пересуватися по екрану.

**Варіант 7:**Вертоліт рухається по екрану в одному напрямку, обидва гвинти крутяться

**Лістинг коду програми**

package main;

import javax.vecmath.\*;

import com.sun.j3d.utils.image.TextureLoader;

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

import javax.media.j3d.\*;

import javax.swing.JFrame;

import com.sun.j3d.loaders.\*;

import com.sun.j3d.loaders.objectfile.\*;

import java.awt.\*;

import java.io.FileNotFoundException;

import java.util.Hashtable;

public class Main extends JFrame{

private static Canvas3D canvas;

private static SimpleUniverse universe;

private static BranchGroup root;

public Main() throws FileNotFoundException {

configureWindow();

configureUniverse();

root = new BranchGroup();

addImageBackground();

addLight();

TransformGroup helicopterTransformGroup = getHelicopterGroup();

root.addChild(helicopterTransformGroup);

root.compile();

universe.addBranchGraph(root);

}

private void configureWindow() {

setTitle("Lab6");

setExtendedState(JFrame.MAXIMIZED\_BOTH);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

private void configureUniverse() {

canvas = new Canvas3D(SimpleUniverse.getPreferredConfiguration());

canvas.setDoubleBufferEnable(true);

getContentPane().add(canvas, BorderLayout.CENTER);

universe = new SimpleUniverse(canvas);

universe.getViewingPlatform().setNominalViewingTransform();

}

private void addImageBackground() {

TextureLoader t = new TextureLoader("D:/projects/maokg/lab6/assets/bg.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);

}

public void addLight(){

BoundingSphere bounds = new BoundingSphere();

bounds.setRadius(100);

DirectionalLight directionalLight = new DirectionalLight(new Color3f(1, 1, 1), new Vector3f(-1, -1, -1));

directionalLight.setInfluencingBounds(bounds);

root.addChild(directionalLight);

AmbientLight ambientLight = new AmbientLight(new Color3f(1, 1, 1));

ambientLight.setInfluencingBounds(new BoundingSphere());

root.addChild(ambientLight);

}

public TransformGroup getHelicopterGroup() throws FileNotFoundException {

ObjectFile loader = new ObjectFile(ObjectFile.LOAD\_ALL);

Scene helicopterScene = loader.load("D:/projects/maokg/lab6/assets/helicopter.obj");

TransformGroup bodyTG = new TransformGroup();

TransformGroup bigPropellerTG = new TransformGroup();

TransformGroup smallPropellerTG = new TransformGroup();

Appearance bodyAppearance = new Appearance();

Color3f col = new Color3f(.23f, .50f, .65f);

bodyAppearance.setMaterial(new Material(col, col, col, col, 150.0f));

Hashtable helicopter = helicopterScene.getNamedObjects();

// helicopter body

Shape3D body = (Shape3D) helicopter.get("body");

Appearance lightRedApp = new Appearance();

setToMyDefaultAppearance(lightRedApp,new Color3f(0,0,0));

body.setAppearance(lightRedApp);

bodyTG.addChild(body.cloneTree());

// big propeller

Shape3D bigPropeller = (Shape3D) helicopter.get("big\_propeller");

bigPropellerTG.addChild(bigPropeller.cloneTree());

// small propeller

Shape3D smallPropeller = (Shape3D) helicopter.get("small\_propeller");

smallPropellerTG.addChild(smallPropeller.cloneTree());

Transform3D startTransformation = new Transform3D();

Transform3D startTransformation2 = new Transform3D();

Transform3D startTransformation3 = new Transform3D();

startTransformation2.rotY(Math.PI / 2);

startTransformation3.rotX(-Math.PI / 8);

startTransformation.setScale(0.3);

startTransformation.mul(startTransformation2);

startTransformation.mul(startTransformation3);

TransformGroup transformGroup = new TransformGroup(startTransformation);

TransformGroup sceneGroup = new TransformGroup();

//

// sceneGroup.addChild(backpackTG);

sceneGroup.addChild(bodyTG);

sceneGroup.addChild(rotateBig(bigPropellerTG));

sceneGroup.addChild(rotateSmall(smallPropellerTG));

transformGroup.addChild(sceneGroup);

return fly(transformGroup);

}

//метод для встановлення матеріалу відповідного кольору

public static void setToMyDefaultAppearance(Appearance app, Color3f col)

{

app.setMaterial(new Material(col,col,col,col,150.0f));

}

private TransformGroup fly(Node node) {

TransformGroup transformGroup = new TransformGroup();

Transform3D tFly = new Transform3D();

tFly.rotZ(-Math.PI);

long flyTime = 10000;

Alpha flyAlpha = new Alpha(1,

Alpha.INCREASING\_ENABLE,

0,

0, flyTime,0,0,0,0,0);

float flyDistance = 0.7f;

PositionInterpolator posIFly = new PositionInterpolator(flyAlpha,

transformGroup,tFly, -0.7f, flyDistance);

BoundingSphere bs = new BoundingSphere(new Point3d(0.0,0.0,0.0),Double.MAX\_VALUE);

posIFly.setSchedulingBounds(bs);

transformGroup.addChild(node);

transformGroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);

transformGroup.addChild(posIFly);

return transformGroup;

}

private TransformGroup rotateBig(Node node){

TransformGroup xformGroup = new TransformGroup();

xformGroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);

Alpha alpha = new Alpha(-1, 500);

Transform3D propellerTranslate = new Transform3D();

Vector3d trans = new Vector3d(0, 0, -0.2);

propellerTranslate.setTranslation(trans);

RotationInterpolator interpolator = new RotationInterpolator(alpha,xformGroup, propellerTranslate,(float) Math.PI\*2,0.0f);

interpolator.setSchedulingBounds(new BoundingSphere(new Point3d(0.0,0.0,0.0),1000));

xformGroup.addChild(interpolator);

xformGroup.addChild(node);

return xformGroup;

}

private TransformGroup rotateSmall(Node node){

TransformGroup xformGroup = new TransformGroup();

xformGroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);

Alpha alpha = new Alpha(-1, 500);

Transform3D propellerTranslate = new Transform3D();

Transform3D propellerRotate = new Transform3D();

Vector3d trans = new Vector3d(0, 0.08, 0.84);

propellerTranslate.setTranslation(trans);

propellerRotate.rotZ(Math.PI/2);

propellerTranslate.mul(propellerRotate);

// propellerTranslate.rotY(Math.PI/2);

RotationInterpolator interpolator = new RotationInterpolator(alpha,xformGroup, propellerTranslate,(float) Math.PI\*2,0.0f);

interpolator.setSchedulingBounds(new BoundingSphere(new Point3d(0.0,0.0,0.0),1000));

xformGroup.addChild(interpolator);

xformGroup.addChild(node);

return xformGroup;

}

public static void main(String[] args) throws FileNotFoundException {

Main window = new Main();

window.setVisible(true);

}

**Результат**

****