

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  Линь Дмитро Іванович  варіант № 10 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною |

Київ 2021

**Варіант завдання**

**Завдання**: Виконати анімацію тривимірної сцени за варіантом

**Обраний варіант даних:** Анімація воїна з сокирою warrior.obj. Воїн повинен взяти в руки сокиру, рухати головою, руками, сокирою

**Лістинг коду програми**

import javax.vecmath.\*;  
  
import com.sun.j3d.utils.image.TextureLoader;  
import com.sun.j3d.utils.universe.\*;  
import javax.media.j3d.\*;  
import com.sun.j3d.utils.behaviors.vp.\*;  
import javax.swing.JFrame;  
import com.sun.j3d.loaders.\*;  
import com.sun.j3d.loaders.objectfile.\*;  
  
import java.io.\*;  
import java.net.URL;  
import java.util.Hashtable;  
import java.util.Enumeration;  
  
public class Main extends JFrame {  
 public Canvas3D myCanvas3D;  
  
 public Main() throws IOException {  
 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("warrior");  
 setSize(900,900);  
 getContentPane().add("Center", myCanvas3D);  
 setVisible(true);  
 }  
  
 public void createSceneGraph(SimpleUniverse su) throws IOException {  
 ObjectFile f = new ObjectFile(ObjectFile.*RESIZE*);  
 BoundingSphere bs = new BoundingSphere(new Point3d(0.0,0.0,0.0),Double.*MAX\_VALUE*);  
 String name;  
 BranchGroup wBGroup = new BranchGroup();  
 Background wBg = new Background(new Color3f(-1.0f,-1.0f,1.0f));  
  
 ClassLoader classLoader = Thread.*currentThread*().getContextClassLoader();  
 InputStream inputStream = classLoader.getResourceAsStream("warrior.obj");  
 Scene warriorScene = f.load(new BufferedReader(new InputStreamReader(inputStream)));  
  
 Hashtable warNamedObjects = warriorScene.getNamedObjects();  
 Enumeration enumer = warNamedObjects.keys();  
  
 while (enumer.hasMoreElements()){  
 name = (String) enumer.nextElement();  
 System.*out*.println("Name: " + name);  
 }  
  
 Transform3D startTransformation = new Transform3D();  
 startTransformation.setScale(2.0/6);  
 Transform3D combinedStartTransformation = new Transform3D();  
 combinedStartTransformation.mul(startTransformation);  
  
 TransformGroup warStartTransformGroup = new TransformGroup(combinedStartTransformation);  
  
 Background background = new Background(getTextureLoader("screenshot1.jpg").getImage());  
 background.setImageScaleMode(Background.*SCALE\_FIT\_MAX*);  
 background.setApplicationBounds(new BoundingSphere(new Point3d(),1000));  
 background.setCapability(Background.*ALLOW\_IMAGE\_WRITE*);  
 warStartTransformGroup.addChild(background);  
  
 int movesCount = -1;  
 int movesDuration = 500;  
 int startTime = 0;  
  
 // appearance  
 Appearance headApp = new Appearance();  
 *setToMyDefaultAppearance*(headApp, new Color3f(0.4f, 0.3f, 0.2f));  
  
 Appearance legsApp = new Appearance();  
 *setToMyDefaultAppearance*(legsApp, new Color3f(0.0f, 0.0f, 0.0f));  
  
 Appearance bodyApp = new Appearance();  
 *setToMyDefaultAppearance*(bodyApp, new Color3f(0.1f, 0.1f, 0.3f));  
  
 Appearance handApp = new Appearance();  
 *setToMyDefaultAppearance*(handApp, new Color3f(0.3f, 0.2f, 0.1f));  
  
 // static rotating  
 Transform3D rightHandRotAxis = new Transform3D();  
 rightHandRotAxis.rotZ(Math.*PI* / 8);  
  
 Transform3D leftHandRotAxis = new Transform3D();  
 rightHandRotAxis.rotZ(Math.*PI* / 8);  
  
 Transform3D headRotAxis = new Transform3D();  
 headRotAxis.rotZ(Math.*PI* / 10);  
  
 Transform3D axeRotStatic = new Transform3D();  
 axeRotStatic.setTranslation(new Vector3f(-0.1f, 0.0f, 0.5f));  
  
 Transform3D axeRotX = new Transform3D();  
 axeRotX.rotX(Math.*PI* / 10);  
  
 Transform3D axeRotY = new Transform3D();  
 axeRotY.rotZ(Math.*PI* / 10);  
 axeRotStatic.mul(axeRotX);  
// axeRotStatic.mul(axeRotY);  
  
 Transform3D axeRotAxis = new Transform3D();  
// axeRotAxis.mul(axeRotStatic);  
  
 // shapes  
  
 Shape3D head = (Shape3D) warNamedObjects.get("head");  
 head.setAppearance(headApp);  
 TransformGroup headTGT = new TransformGroup();  
 TransformGroup headTG = rotate(headTGT, new Alpha(1,2000));  
 headTG.addChild(head.cloneTree());  
  
 Shape3D rightHand = (Shape3D) warNamedObjects.get("right\_hand");  
 rightHand.setAppearance(bodyApp);  
 TransformGroup rightHandTG = new TransformGroup();  
 rightHandTG.addChild(rightHand.cloneTree());  
  
 Shape3D leftHand = (Shape3D) warNamedObjects.get("left\_hand");  
 leftHand.setAppearance(bodyApp);  
 TransformGroup leftHandTG = new TransformGroup();  
 leftHandTG.addChild(leftHand.cloneTree());  
  
 Shape3D axe = (Shape3D) warNamedObjects.get("box02\_group1");  
 TransformGroup axeTG = new TransformGroup();  
 axeTG.setTransform(axeRotStatic);  
 axeTG.addChild(axe.cloneTree());  
  
 Shape3D body = (Shape3D) warNamedObjects.get("group1\_\_\_\_\_02");  
 TransformGroup bodyTG = new TransformGroup();  
 body.setAppearance(bodyApp);  
 bodyTG.addChild(body.cloneTree());  
  
 Shape3D legs = (Shape3D) warNamedObjects.get("group1\_\_\_\_\_01");  
 TransformGroup legsTG = new TransformGroup();  
 legs.setAppearance(legsApp);  
 legsTG.addChild(legs.cloneTree());  
  
 // permanent animations  
  
 Alpha alpha = new Alpha(movesCount, Alpha.*INCREASING\_ENABLE*, startTime, 0, movesDuration,0,0,0,0,0);  
  
 RotationInterpolator handRightRot = new RotationInterpolator(alpha, rightHandTG, rightHandRotAxis, (float) Math.*PI*/10,(float) -Math.*PI*/10);  
 handRightRot.setSchedulingBounds(bs);  
 rightHandTG.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  
 rightHandTG.addChild(handRightRot);  
  
 RotationInterpolator handLeftRot = new RotationInterpolator(alpha, leftHandTG, leftHandRotAxis, (float) Math.*PI*/10,(float) -Math.*PI*/10);  
 handLeftRot.setSchedulingBounds(bs);  
 leftHandTG.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  
 leftHandTG.addChild(handLeftRot);  
  
 RotationInterpolator headRot = new RotationInterpolator(alpha, headTG, headRotAxis, (float) Math.*PI*/5,(float) Math.*PI*/10);  
 headRot.setSchedulingBounds(bs);  
 headTG.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  
 headTG.addChild(headRot);  
  
 RotationInterpolator axeRot = new RotationInterpolator(alpha, axeTG, axeRotAxis, (float) 0.0f,(float) (-Math.*PI* / 4));  
 axeRot.setSchedulingBounds(bs);  
 axeTG.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  
 axeTG.addChild(axeRot);  
  
 // body  
 TransformGroup sceneGroup = new TransformGroup();  
 sceneGroup.addChild(headTG);  
 sceneGroup.addChild(rightHandTG);  
 sceneGroup.addChild(leftHandTG);  
 sceneGroup.addChild(axeTG);  
 sceneGroup.addChild(bodyTG);  
 sceneGroup.addChild(legsTG);  
  
 TransformGroup whiteTransXformGroup = translate(  
 warStartTransformGroup,  
 new Vector3f(0.0f,0.0f,0.5f));  
  
 wBGroup.addChild(whiteTransXformGroup);  
 warStartTransformGroup.addChild(sceneGroup);  
  
 BoundingSphere bounds = new BoundingSphere(new Point3d(120.0,250.0,100.0),Double.*MAX\_VALUE*);  
 wBg.setApplicationBounds(bounds);  
 wBGroup.addChild(wBg);  
  
 wBGroup.compile();  
 su.addBranchGraph(wBGroup);  
 }  
  
 private TextureLoader getTextureLoader(String path) throws IOException {  
 ClassLoader classLoader = Thread.*currentThread*().getContextClassLoader();  
 URL textureResource = classLoader.getResource(path);  
 if (textureResource == null) {  
 throw new IOException("Couldn't find texture: " + path);  
 }  
 return new TextureLoader(textureResource.getPath(), myCanvas3D);  
 }  
  
 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);  
 }  
  
 private TransformGroup translate(Node node, Vector3f vector){  
  
 Transform3D transform3D = new Transform3D();  
 transform3D.setTranslation(vector);  
 TransformGroup transformGroup =  
 new TransformGroup();  
 transformGroup.setTransform(transform3D);  
  
 transformGroup.addChild(node);  
 return transformGroup;  
 }  
  
 private TransformGroup rotate(Node node, Alpha alpha){  
 TransformGroup xformGroup = new TransformGroup();  
 xformGroup.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  
  
 RotationInterpolator interpolator =  
 new RotationInterpolator(alpha,xformGroup);  
  
 interpolator.setSchedulingBounds(new BoundingSphere(  
 new Point3d(0.0,0.0,0.0),1.0));  
  
 xformGroup.addChild(interpolator);  
 xformGroup.addChild(node);  
  
 return xformGroup;  
 }  
  
 public static void setToMyDefaultAppearance(Appearance app, Color3f col) {  
 app.setMaterial(new Material(col, col, col, col, 150.0f));  
 }  
  
 public static void main(String[] args) throws IOException {  
 Main m = new Main();  
 }  
}

**Результат у директорії разом з цим документом**

Файл у форматі gif