

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 теми “Анімація тривимірних об’єктів ”

|  |  |  |  |
| --- | --- | --- | --- |
| Виконала  студентка III курсу  групи КП-82  Лахман Ксенія  (*прізвище, ім’я, по батькові*)  варіант № 11 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2021

**Мета:**

Навчитися анімувати складні об’єкти тривимірної сцени.

**Завдання**:

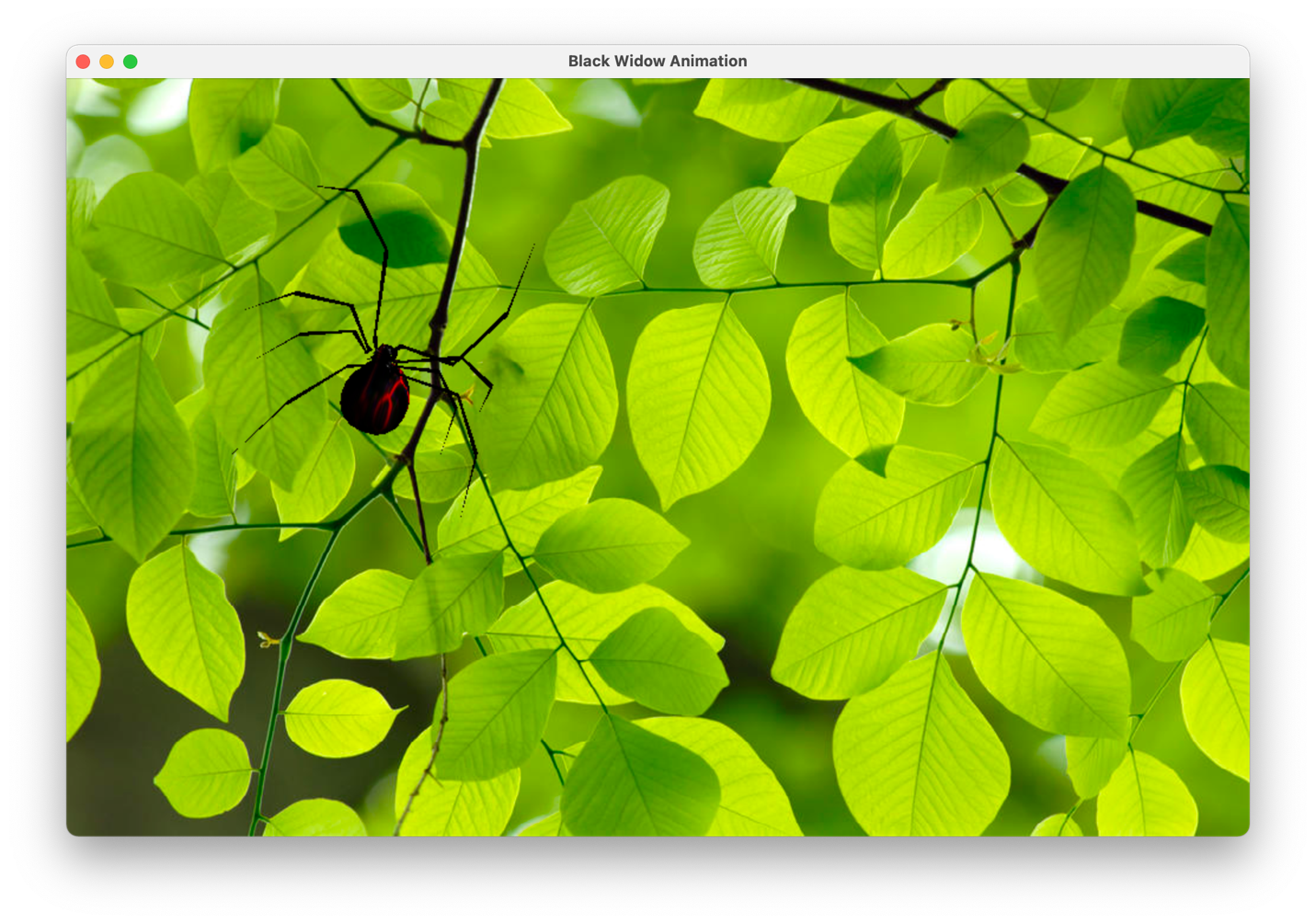
Виконати анімацію тривимірної сцени за варіантом.

Варіант: 1 Анімація павука «чорна вдова» black\_widow.obj. Рух ніжок, пересування по екрану.

**Лістинг коду програми**

|  |
| --- |
| BlackWidowAnimation.java |
| import com.sun.j3d.loaders.Scene; import com.sun.j3d.loaders.objectfile.ObjectFile; import com.sun.j3d.utils.behaviors.vp.OrbitBehavior; import com.sun.j3d.utils.image.TextureLoader; import com.sun.j3d.utils.universe.SimpleUniverse;  import javax.media.j3d.\*; import javax.swing.\*; import javax.vecmath.Color3f; import javax.vecmath.Point3d; import javax.vecmath.Vector3f; import java.awt.\*; import java.util.Enumeration; import java.util.Hashtable;  public class BlackWidowAnimation extends JFrame {  public Canvas3D myCanvas3D;  public BranchGroup theScene;  public BoundingSphere bs;  Hashtable roachNamedObjects;   int noRotHour = 100; *//кількість обертів* int timeRotationHour = 300;*//час одного оберту* public BlackWidowAnimation() {  theScene = new BranchGroup();  this.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);   myCanvas3D = new Canvas3D(SimpleUniverse.*getPreferredConfiguration*());   theScene.setCapability(BranchGroup.*ALLOW\_CHILDREN\_EXTEND*);  addImageBackground("resources/image/pic.jpg");   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);   configureWindow();  }   public static void main(String[] args) {  BlackWidowAnimation blackWidowAnimation = new BlackWidowAnimation();  }   private void configureWindow() {  setTitle("Black Widow Animation");  setSize(1000, 669);  getContentPane().add("Center", myCanvas3D);  setResizable(true);  setVisible(true);  setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  }   private void addImageBackground(String path) {  TextureLoader t = new TextureLoader(path, myCanvas3D);  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);  theScene.addChild(background);  }   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 void createSceneGraph(SimpleUniverse su) {  ObjectFile f = new ObjectFile(ObjectFile.*RESIZE*);  Scene widowScene = null;  try {  widowScene = f.load("resources/black\_widow.obj");  } catch (Exception e) {  System.*out*.println("File loading failed:" + e);  }   Transform3D startTransformation = new Transform3D();  startTransformation.setScale(1.0 / 4);  Transform3D tfRoach = new Transform3D();  tfRoach.rotY(-3\*Math.*PI*/2);  tfRoach.mul(startTransformation);  *//* TransformGroup tgRoach = new TransformGroup(tfRoach);  TransformGroup sceneGroup = new TransformGroup();  *//* roachNamedObjects = widowScene.getNamedObjects();  Enumeration enumer = roachNamedObjects.keys();  String name = "";  while (enumer.hasMoreElements()) {  name = (String) enumer.nextElement();  System.*out*.println("Name: " + name);  }   TransformGroup tgBody = new TransformGroup();  Shape3D body\_widow = (Shape3D) roachNamedObjects.get("blkw\_body");  body\_widow.setAppearance(loadTexture("resources/image/skin.jpg"));  tgBody.addChild(body\_widow.cloneTree());  sceneGroup.addChild(tgBody.cloneTree());   Appearance app = new Appearance();  Color3f black = new Color3f(Color.*BLACK*);  app.setMaterial(new Material(black, black, black, black, 0f));   *// Animation* bs = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), Double.*MAX\_VALUE*);   TransformGroup tgLeg1\_1 = createLeg("leg1", app, 0, (float) Math.*PI* / 2, true);  TransformGroup tgLeg2\_1 = createLeg("leg2", app, 100, (float) Math.*PI* / 8, false);  TransformGroup tgLeg3\_1 = createLeg("leg3", app, 200, (float) Math.*PI* / 8, false);  TransformGroup tgLeg4\_1 = createLeg("leg4", app, 300, (float) Math.*PI* / 8, false);  TransformGroup tgLeg1\_2 = createLeg("leg8", app, 200, (float) Math.*PI* / 2, true);  TransformGroup tgLeg2\_2 = createLeg("leg7", app, 300, -(float) Math.*PI* / 8, false);  TransformGroup tgLeg3\_2 = createLeg("leg6", app, 400, -(float) Math.*PI* / 8, false);  TransformGroup tgLeg4\_2 = createLeg("leg5", app, 500, -(float) Math.*PI* / 8, false);   sceneGroup.addChild(tgLeg1\_1);  sceneGroup.addChild(tgLeg2\_1);  sceneGroup.addChild(tgLeg3\_1);  sceneGroup.addChild(tgLeg4\_1);  sceneGroup.addChild(tgLeg1\_2);  sceneGroup.addChild(tgLeg2\_2);  sceneGroup.addChild(tgLeg3\_2);  sceneGroup.addChild(tgLeg4\_2);   sceneGroup.addChild(tgBody.cloneTree());  *//* TransformGroup whiteTransXformGroup = translate(  tgRoach,  new Vector3f(0.0f,0.0f,0.5f));   TransformGroup whiteRotXformGroup = rotate(whiteTransXformGroup, new Alpha(10,5000));  theScene.addChild(whiteRotXformGroup);  tgRoach.addChild(sceneGroup);  *//* theScene.compile();  su.addBranchGraph(theScene);  }   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;  }   TransformGroup rotate(Node node,Alpha alpha){  TransformGroup xformGroup = new TransformGroup();  xformGroup.setCapability(  TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);   *//Create an interpolator for rotating the node.* RotationInterpolator interpolator =  new RotationInterpolator(alpha,xformGroup);   *//Establish the animation region for this  // interpolator.* interpolator.setSchedulingBounds(new BoundingSphere(  new Point3d(0.0,0.0,0.0),1.0));   *//Populate the xform group.* xformGroup.addChild(interpolator);  xformGroup.addChild(node);   return xformGroup;  }   public TransformGroup createLeg(String objectName, Appearance app, long triggerTime, float rotationVelocity, boolean isMain) {  Alpha legRotAlpha = new Alpha(noRotHour, Alpha.*INCREASING\_ENABLE*, triggerTime, 0, timeRotationHour,  0, 0, 0, 0, 0);   Shape3D leg = (Shape3D) roachNamedObjects.get(objectName);  leg.setAppearance(app);  TransformGroup tgLeg = new TransformGroup();  tgLeg.addChild(leg.cloneTree());   Transform3D legRotAxis = new Transform3D();  if (isMain) {  legRotAxis.rotZ(Math.*PI*/2);  }  RotationInterpolator legRotation = new RotationInterpolator(legRotAlpha, tgLeg, legRotAxis, rotationVelocity, 0.0f);  legRotation.setSchedulingBounds(bs);  tgLeg.setCapability(TransformGroup.*ALLOW\_TRANSFORM\_WRITE*);  tgLeg.addChild(legRotation);   return tgLeg;  }   Appearance loadTexture(String path) {  Texture tex = new TextureLoader(path, this).getTexture();  tex.setBoundaryModeS(Texture.*WRAP*);  tex.setBoundaryModeT(Texture.*WRAP*);   TextureAttributes texAttr = new TextureAttributes();  texAttr.setTextureMode(TextureAttributes.*COMBINE*);   Appearance ap = new Appearance();  ap.setTexture(tex);  ap.setTextureAttributes(texAttr);   Material material = new Material();  material.setSpecularColor(new Color3f(Color.*WHITE*));  material.setDiffuseColor(new Color3f(Color.*WHITE*));  ap.setMaterial(material);   return ap;  } } |

**Результат**

****