

МIНIСТЕРСТВО ОСВIТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота № 5**

з дисципліни “Математичні та алгоритмічні основи компютерної графіки”

тема «Імпорт тривімирних моделей у середовище програмування java3D,

обробка та маніапуляція цих зображень»

|  |  |  |
| --- | --- | --- |
| Виконав:  студент III курсу  групи КП-71  Мікулін Володимир Олександрович  варіант № 13  Сумарний бал: |  | Перевірила  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_”  2020р.  викладач  Шкурат Оксана Сергіївна |
|  |  |  |

|  |  |
| --- | --- |
|  |  |

Київ 2020

**Мета**:

1. Здобути навички імпорту моделей, побудованих у тривимірних редакторах,

(об’єктів форматів .obj, .lwo, .3ds) до бібліотек.

**Завдання на лабораторну роботу**

Імпортувати моделі тривимірних об’єктів форматів, що визначені варіантом.

Створити реалістичну анімацію об’єкту. Додати до сцени фон, інші об’єкти для

надання сцені реалістичного вигляду. Для цього використати текстури,

матеріали, імпортувати додаткові об’єкти з відкритих бібліотек, за бажанням

створити прості об’єкти у графічному редакторі.

Студенти, які мають непарний номер варіанту у списку групи імпортують

моделі формату .obj, парний варіант – .lwo.

**Варіант 13:** Не парний.

Використав приклад наведений у завданні до лабораторної із прикладів сцен для анімації.

“Імпортувати модель паперового кораблика. Анімувати рух кораблика по

весняній калюжі.”

**Текст коду програм**

|  |
| --- |
| **AnimationBoat**.java |
| public class AnimationBoat implements ActionListener {  private Button go;  private TransformGroup wholePlane;  private Transform3D translateTransform;  private Transform3D rotateTransformX;  private Transform3D rotateTransformY;  private Transform3D rotateTransformZ;    private JFrame mainFrame;    private float sign=1.0f;  private float zoom=1.0f;  private float xloc=0.0f;  private float yloc=-1.3f;  private float zloc=0.0f;  private int moveType=1;  private Timer timer;    public AnimationBoat(TransformGroup wholeBoat, Transform3D trans, JFrame frame){  go = new Button("Go");  this.wholePlane=wholeBoat;  this.translateTransform=trans;  this.mainFrame=frame;    rotateTransformX= new Transform3D();  rotateTransformY= new Transform3D();  rotateTransformZ= new Transform3D();  timer = new Timer(100, this);    Panel p =new Panel();  p.add(go);  mainFrame.add("North",p);  go.addActionListener(this);  }  private void initialBoteState(){  xloc=0.0f;  yloc=0.0f;  zloc=0.0f;  zoom=1.0f;  moveType=1;  sign=1.0f;  if(timer.isRunning()){timer.stop();}  go.setLabel("Go");  }    @Override  public void actionPerformed(ActionEvent e) {  // start timer when button is pressed  if (e.getSource()==go){  if (!timer.isRunning()) {  timer.start();  go.setLabel("Stop");  }  else {  timer.stop();  go.setLabel("Go");  }  }  else {  Move(moveType);  translateTransform.setScale(new Vector3d(zoom, zoom, zoom));  translateTransform.setTranslation(new Vector3f(xloc,yloc,zloc));  wholePlane.setTransform(translateTransform);  }  }  private void Move(int mType){  if(mType==1){  xloc += 0.1 \* sign;  if (Math.abs(xloc \*2) >= 2 ) {  sign = -1.0f \* sign;  rotateTransformZ.rotZ(Math.PI);  translateTransform.mul(rotateTransformZ);  }  }  if(mType == 2){  if(zoom < 0.03){  initialBoteState();  }  else{  xloc += 0.04;  yloc += 0.003;  zloc -= 0.005;  zoom -= 0.005;  }  }  }  } |

|  |
| --- |
| **Main**.java |
| public class Main extends JFrame {  static SimpleUniverse universe;  static Scene scene;  static Map<String, Shape3D> nameMap;  static BranchGroup root;  static Canvas3D canvas;    static TransformGroup wholeBoat;  static Transform3D transform3D;    public Main() throws IOException{  configureWindow();  configureCanvas();  configureUniverse();  addModelToUniverse();  setPlaneElementsList();  addAppearance();  addImageBackground();  addLightToUniverse();  addOtherLight();  ChangeViewAngle();  root.compile();  universe.addBranchGraph(root);  }    private void configureWindow() {  setTitle("Boat Animation");  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("source\_folder//cat\_boat.obj");  root=scene.getSceneGroup();  }    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);  }    private void printModelElementsList(Map<String,Shape3D> nameMap){  for (String name : nameMap.keySet()) {  System.out.printf("Name: %s\n", name);}  }    private void setPlaneElementsList() {  nameMap=scene.getNamedObjects();  //Print elements of your model:  printModelElementsList(nameMap);    wholeBoat = new TransformGroup();  transform3D = new Transform3D();  transform3D.rotX(-Math.PI / 2);  wholeBoat.setTransform(transform3D);  transform3D.setTranslation(new Vector3f(0, -1.3f, 0));  wholeBoat.setTransform(transform3D);  root.removeChild(nameMap.get("19292\_cat\_boat\_v1"));  wholeBoat.addChild(nameMap.get("19292\_cat\_boat\_v1"));  wholeBoat.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);  root.addChild(wholeBoat);  }    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( 0.9f, 0.9f, 0.9f) );  material.setDiffuseColor ( new Color3f( 1f, 1f, 1f ) );  material.setSpecularColor( new Color3f( 1f, 1f, 1f ) );  material.setShininess( 0.3f );  material.setLightingEnable(true);  return material;  }    private void addAppearance(){  Appearance boatAppearance = new Appearance();  boatAppearance.setTexture(getTexture("source\_folder//paper.jpg"));  TextureAttributes texAttr = new TextureAttributes();  texAttr.setTextureMode(TextureAttributes.COMBINE);  boatAppearance.setTextureAttributes(texAttr);  boatAppearance.setMaterial(getMaterial());  Shape3D plane = nameMap.get("19292\_cat\_boat\_v1");  plane.setAppearance(boatAppearance);  }    private void addColorBackground(){  Background background = new Background(new Color3f(Color.CYAN));  BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0),100.0);  background.setApplicationBounds(bounds);  root.addChild(background);  }    private void addImageBackground(){  TextureLoader t = new TextureLoader("source\_folder//lake.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 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);  }    private void addOtherLight(){  Color3f directionalLightColor = new Color3f(Color.BLACK);  Color3f ambientLightColor = new Color3f(Color.WHITE);  Vector3f lightDirection = new Vector3f(-1F, -1F, -1F);  AmbientLight ambientLight = new AmbientLight(ambientLightColor);  DirectionalLight directionalLight = new DirectionalLight(directionalLightColor, lightDirection);  Bounds influenceRegion = new BoundingSphere();  ambientLight.setInfluencingBounds(influenceRegion);  directionalLight.setInfluencingBounds(influenceRegion);  root.addChild(ambientLight);  root.addChild(directionalLight);  }    public static Scene getSceneFromFile(String location) throws IOException {  ObjectFile file = new ObjectFile(ObjectFile.RESIZE);  file.setFlags (ObjectFile.RESIZE | ObjectFile.TRIANGULATE | ObjectFile.STRIPIFY);  return file.load(new FileReader(location));  }    //Not always works  public static Scene getSceneFromLwoFile(String location) throws IOException {  Lw3dLoader loader = new Lw3dLoader();  return loader.load(new FileReader(location));  }    public static void main(String[]args){  try {  Main window = new Main();  AnimationBoat planeMovement = new AnimationBoat(wholeBoat, transform3D, window);  window.setVisible(true);  }  catch (IOException ex) {  System.out.println(ex.getMessage());  }  }  } |

**Висновки**

Виконавши дану лабораторну роботу я здобув навички імпорту моделей, побудованих у тривимірних редакторах, об’єктів форматів .obj до бібліотек.