



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”  
тема “Анімація тривимірних об’єктів”

Виконав(ла)  
студент(ка) III курсу  
групи КП-83  
Матіюк Дарина Андріївна

Зарахована  
“ \_\_\_\_ ” “ \_\_\_\_\_ ” 20\_\_ р.  
Шкурат Оксаною Сергіївною

варіант №12

Київ 2021

**Варіант завдання**

**Завдання:** виконати анімацію тривимірної сцени за варіантом.

**Варіант:** анімація автомобіля car.obj. Рух коліс, пересування по екрану, обов'язкові повороти авто.

## Лістинг коду програми

### Main.java

```

public class Car extends JFrame {
    public Canvas3D canvas;
    public Car() throws IOException {
        // canvas & universe
        canvas = new Canvas3D(SimpleUniverse.getPreferredConfiguration());
        SimpleUniverse universe = new SimpleUniverse(canvas);
        universe.getViewingPlatform().setNominalViewingTransform();
        createSceneGraph(universe);
        // window
        setTitle("lab6");
        setSize(800, 600);
        getContentPane().add("Center", canvas);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        // mouse navigation
        OrbitBehavior ob = new OrbitBehavior(canvas);
        ob.setSchedulingBounds(new BoundingSphere(new
Point3d(0.0,0.0,0.0),Double.MAX_VALUE));
        universe.getViewingPlatform().setViewPlatformBehavior(ob);
        // light
        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);
        universe.addBranchGraph(bgLight);
    }
    public static void main(String[] args) throws IOException {
        Car car = new Car();
    }
    public void createSceneGraph(SimpleUniverse universe) throws IOException {
        BoundingSphere bounds = new BoundingSphere(new
Point3d(0.0,0.0,0.0),Double.MAX_VALUE);
        ObjectFile file = new ObjectFile(ObjectFile.RESIZE);
        TextureLoader tl = new TextureLoader("assets/back.jpg", canvas);
        Background back = new Background(tl.getImage());
        BranchGroup carBG = new BranchGroup();
        Scene carScene = null;
        try {
            carScene = file.load("assets/car.obj");
        } catch (Exception e) {
            System.out.println("File loading failed ->" + e);
        }
        Transform3D tfCar = new Transform3D();
        tfCar.rotZ(0);
        tfCar.rotY(Math.PI/3);
        tfCar.setScale(1.0/4);
        TransformGroup tgCar = new TransformGroup(tfCar);
        Hashtable carNamedObjects = carScene.getNamedObjects();
        //paint car to orange
        Appearance redCarA = new Appearance();
    }
}

```

```

        setToMyDefaultAppearance(redCarA, new Color3f(0.8f, 0.1f, 0.0f));
        Shape3D redCar = (Shape3D) carNamedObjects.get("car");
        redCar.setAppearance(redCarA);
        Shape3D wheel4 = (Shape3D) carNamedObjects.get("wheel4");
        Shape3D wheel3 = (Shape3D) carNamedObjects.get("wheel3");
        Shape3D wheel2 = (Shape3D) carNamedObjects.get("wheel2");
        Shape3D wheel1 = (Shape3D) carNamedObjects.get("wheel1");
        Shape3D[] car = new Shape3D[] { redCar };
        for (Shape3D shape:car) {
            tgCar.addChild(shape.cloneTree());
        }
        Transform3D startTransformation = new Transform3D();
        Transform3D combinedStartTransformation = new Transform3D();
        combinedStartTransformation.mul(startTransformation);
        TransformGroup carStartTransformGroup = new
TransformGroup(combinedStartTransformation);
        // animation of wheels
        int timeStart = 1000; // time for animation to start
        int numRot = 100; // number of rotations
        int timeRot = 3600; // time of 1 rotation
        Transform3D wheel4RotAxis = new Transform3D();
        wheel4RotAxis.set(new Vector3d(0, -0.101, 0.55));
        wheel4RotAxis.setRotation(new AxisAngle4d(0, 0, -0.1, Math.PI / 2));
        TransformGroup tgWheel4 = new TransformGroup();
        tgWheel4.addChild(wheel4.cloneTree());
        Transform3D wheel3RotAxis = new Transform3D();
        wheel3RotAxis.set(new Vector3d(0, -0.101, -0.6));
        wheel3RotAxis.setRotation(new AxisAngle4d(0, 0, -0.1, Math.PI / 2));
        TransformGroup tgWheel3 = new TransformGroup();
        tgWheel3.addChild(wheel3.cloneTree());
        Transform3D wheel2RotAxis = new Transform3D();
        wheel2RotAxis.set(new Vector3d(0, -0.095, 0.5));
        wheel2RotAxis.setRotation(new AxisAngle4d(0, 0, -0.1, Math.PI / 2));
        TransformGroup tgWheel2 = new TransformGroup();
        tgWheel2.addChild(wheel2.cloneTree());
        Transform3D wheel1RotAxis = new Transform3D();
        wheel1RotAxis.set(new Vector3d(0, -0.095, -0.65));
        wheel1RotAxis.setRotation(new AxisAngle4d(0, 0, -0.1, Math.PI / 2));
        TransformGroup tgWheel1 = new TransformGroup();
        tgWheel1.addChild(wheel1.cloneTree());
        Alpha wheelRotAlpha = new Alpha(numRot, Alpha.INCREASING_ENABLE,
timeStart, 0, timeRot, 0, 0, 0, 0);
        // wheel4
        RotationInterpolator wheel4Rot = new
RotationInterpolator(wheelRotAlpha, tgWheel4, wheel4RotAxis, 0.0f, (float)
Math.PI * 2);
        wheel4Rot.setSchedulingBounds(bounds);
        tgWheel4.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        tgWheel4.addChild(wheel4Rot);
        // wheel3
        RotationInterpolator wheel3Rot = new
RotationInterpolator(wheelRotAlpha, tgWheel3, wheel3RotAxis, 0.0f, (float)
Math.PI * 2);
        wheel3Rot.setSchedulingBounds(bounds);
        tgWheel3.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        tgWheel3.addChild(wheel3Rot);
        // wheel2
        RotationInterpolator wheel2Rot = new
RotationInterpolator(wheelRotAlpha, tgWheel2, wheel2RotAxis, 0.0f, (float)
Math.PI * 2);

```

```

        wheel2Rot.setSchedulingBounds(bounds);
        tgWheel2.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        tgWheel2.addChild(wheel2Rot);
        // wheel1
        RotationInterpolator wheel1Rot = new
RotationInterpolator(wheelRotAlpha, tgWheel1, wheel1RotAxis, 0.0f, (float)
Math.PI * 2);
        wheel1Rot.setSchedulingBounds(bounds);
        tgWheel1.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
        tgWheel1.addChild(wheel1Rot);
        // end of animation
        Transform3D tfWheel = new Transform3D();
        tfWheel.rotY(Math.PI/3);
        tfWheel.setScale(1.0/4);
        TransformGroup tgCarWheel4 = new TransformGroup(tfWheel);
        tgCarWheel4.addChild(tgWheel4);
        TransformGroup tgCarWheel3 = new TransformGroup(tfWheel);
        tgCarWheel3.addChild(tgWheel3);
        TransformGroup tgCarWheel2 = new TransformGroup(tfWheel);
        tgCarWheel2.addChild(tgWheel2);
        TransformGroup tgCarWheel1 = new TransformGroup(tfWheel);
        tgCarWheel1.addChild(tgWheel1);
        BranchGroup theScene = new BranchGroup();
        theScene.addChild(tgCar);
        theScene.addChild(tgCarWheel4);
        theScene.addChild(tgCarWheel3);
        theScene.addChild(tgCarWheel2);
        theScene.addChild(tgCarWheel1);
        TransformGroup whiteTransXformGroup = translate(carStartTransformGroup,
new Vector3f(0.0f,0.0f,0.5f));
        TransformGroup whiteRotXformGroup = rotate(whiteTransXformGroup, new
Alpha(10,5000));
        carBG.addChild(whiteRotXformGroup);
        carStartTransformGroup.addChild(theScene);
        // add background
        back.setImageScaleMode(Background.SCALE_FIT_MAX);
        back.setApplicationBounds(bounds);
        back.setCapability(Background.ALLOW_IMAGE_WRITE);
        theScene.addChild(back);
        carBG.compile();
        universe.addBranchGraph(carBG);
    }
    public static void setToMyDefaultAppearance(Appearance app, Color3f col){
        app.setMaterial(new Material(col,col,col,col,150.0f));
    }
    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;  
}  
}
```

**Результат**



*Рис. 1-3. Результати роботи програми.*