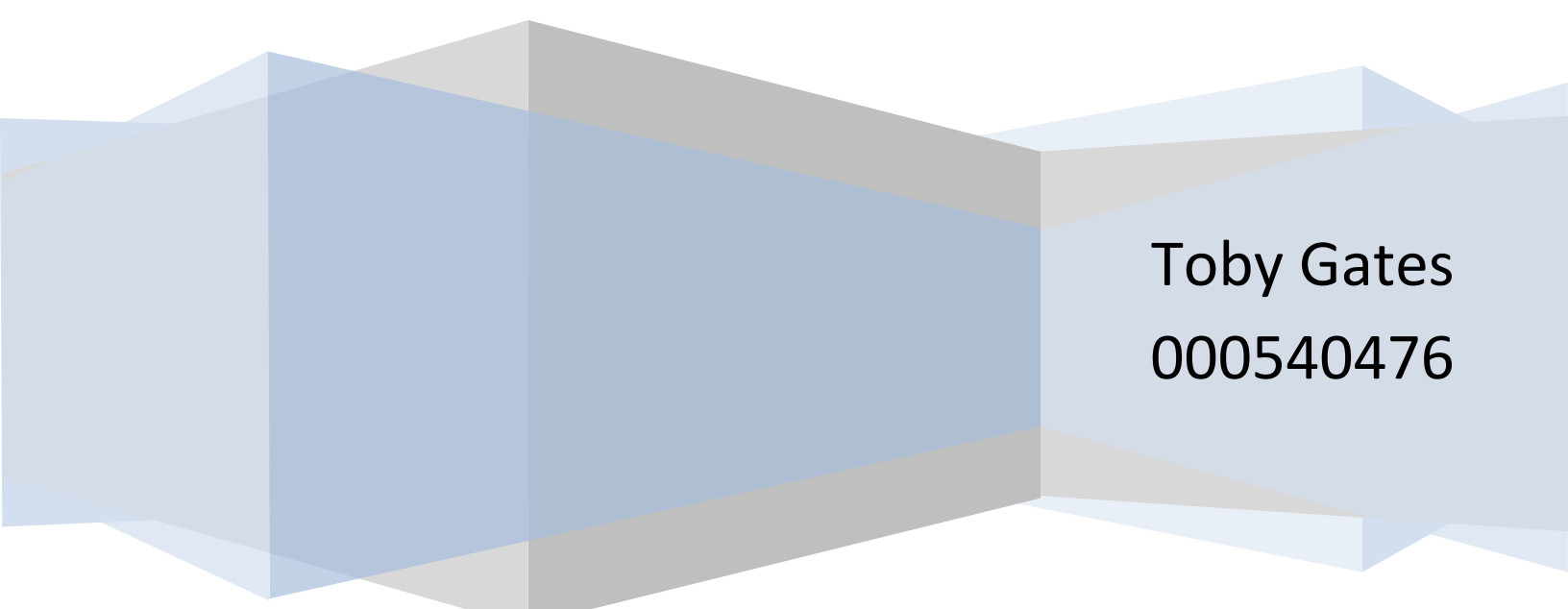


# 3D Graphics (Coursework 2)

Java 3D



Toby Gates  
000540476

## Table of Contents

How the scene was designed.....	3
Description of User Interaction.....	3
Critical analysis of the coursework, a discussion of what you would do differently if you were to attempt the coursework again and another informative discussion of any difficulties encountered .....	5
Discussion of the Highs and Lows of scene-graph approach (Java 3D) opposed to the state-machine based approach (JOGL) .....	5
Appendix A – Scene Graph of the Code .....	6
Appendix B – Screenshots of the Scene.....	7
Appendix C – Source Code for the Robot .....	11

## How the scene was designed

The scene was initially designed with idea of solely using mouse clicks on a screen with the robot along with the use of picking features; but seeing as how a mouse only has three buttons and picking wouldn't provide much variation for the user, it was redesigned.

The same initial idea provided before including the robot was implemented but placed in a separate screen to the right; this version now features added buttons to the left hand side of the screen for various options. These buttons are split into two sections to provide an easy way of helping the user understand what they are for, there is also text based help explaining what each button does as well as the general features found in this program applet.

It was necessary for to allow the user to be able to edit the scene with both picking options and just general interaction.

## Description of User Interaction

The User Interaction in my coursework is split up into three areas; two sections of buttons for the user to press and one which involves heavy use of the mouse. The first section of buttons are various options for picking for when the user has played around with picking on the robot, the second section of buttons for scene interaction such as the toggling of lights for example and the last bit of interaction that involves heavy use of the mouse is the picking itself.

### Picking Options - Buttons

Listed below are the descriptions of the picking feature buttons, note that picking has to be done in order to see the effects of each on the robot. If you press one and see nothing it's because you simply haven't used picking not because it didn't take effect.

**Squares** – Squares is a feature that shows the picked objects as normal picked objects however the edges of the shapes are outlined with tiny squares on the corner of each side that has picking applied to it. When multiple shapes have been picked you can see multiple outlines indicated with squares and how they all connect together.

**Wireframe** – Wireframe displays the model as a non coloured set of objects with the appearance of shapes made out of wire instead of solid objects like the normal appearance.

**Line** – Line displayed the model as a non coloured set of objects as well and is similar to wireframe except instead of being a stationary wireframe object, the lines are made of little sections which rotate making it look like each line is moving in a certain direction.

**Shadow** – Shadow is an option that makes it look similar to its default coloured setting except shadow displays it all in one colour and looks like it gives the appearance of a shadow or silhouette due to the colour against the background.

### **Interaction - Buttons**

**Hide Robot** – This button simply hides the robot from view; although it is hidden the rotation will still apply and the robot will be facing the direction the rotation has reached when the button below is pressed.

**Show Robot** – This button shows the robot at the point it's reached during a rotation; the robot will only show and this feature will only work if the robot had been hidden with the button above.

### **Picking – Mouse Input**

**Left Click** – When the mouse is left clicked when hovering over a part of the robot, the selected part of the robot turns white, if a Picking Option button is/has been pressed then the effects of the picking and appearance of the robot changes to the picking option selected.

**Note:** this only works when highlighting part of the robot because when any other area is clicked in the applet other possible effects may take place such as button presses. But usually nothing may happen depending on the area that was clicked.

### **General Mouse Input**

**Left Click** – Left Clicking allows the Rotation of the Robot when part of the robot is not highlighted meaning you could rotate the robot upside down or rotate it to face a different angle if you wanted.

**Right Click** – Holding the Right Mouse button and moving the mouse allows the translation of the robot thus allowing the user to move it to wherever they want on the screen.

**Mouse Wheel** – When the Mouse Wheel is scrolled up the screen zooms in, when the mouse wheel is scrolled down the screen zooms out. Zooming can also be done by clicking the Mouse Wheel and holding it then moving the mouse either up or down to zoom in or zoom out respectively.

## **Critical analysis of the coursework, a discussion of what you would do differently if you were to attempt the coursework again and another informative discussion of any difficulties encountered**

I believe my coursework was completed to a decent standard with a suitable amount of features implemented and so forth. Despite this I still feel there is a lot of room for improvements and modifications to be done to the program, many adjustments would probably be made if I had to do it all again.

If I were to do the coursework again, the approach taken would've been a little different. Instead of designing a robot and then implementing the features afterwards; I probably would've developed a couple of the features together with building the robot so I would save time and achieve things faster and give myself more time to focus on Animation or another additional feature later on.

I would've also drawn out a draft Scene Graph or set of UML diagrams for the program before coding anything in order to have a base foundation of how the program would be laid out, this would've allowed the final Scene Graph to be improved upon during development and completed to a good standard by submission.

Overall, there weren't any major difficulties encountered. However I did have trouble with removing a child from a Branch Group and still didn't manage to work out how to do it if it's even possible to do.

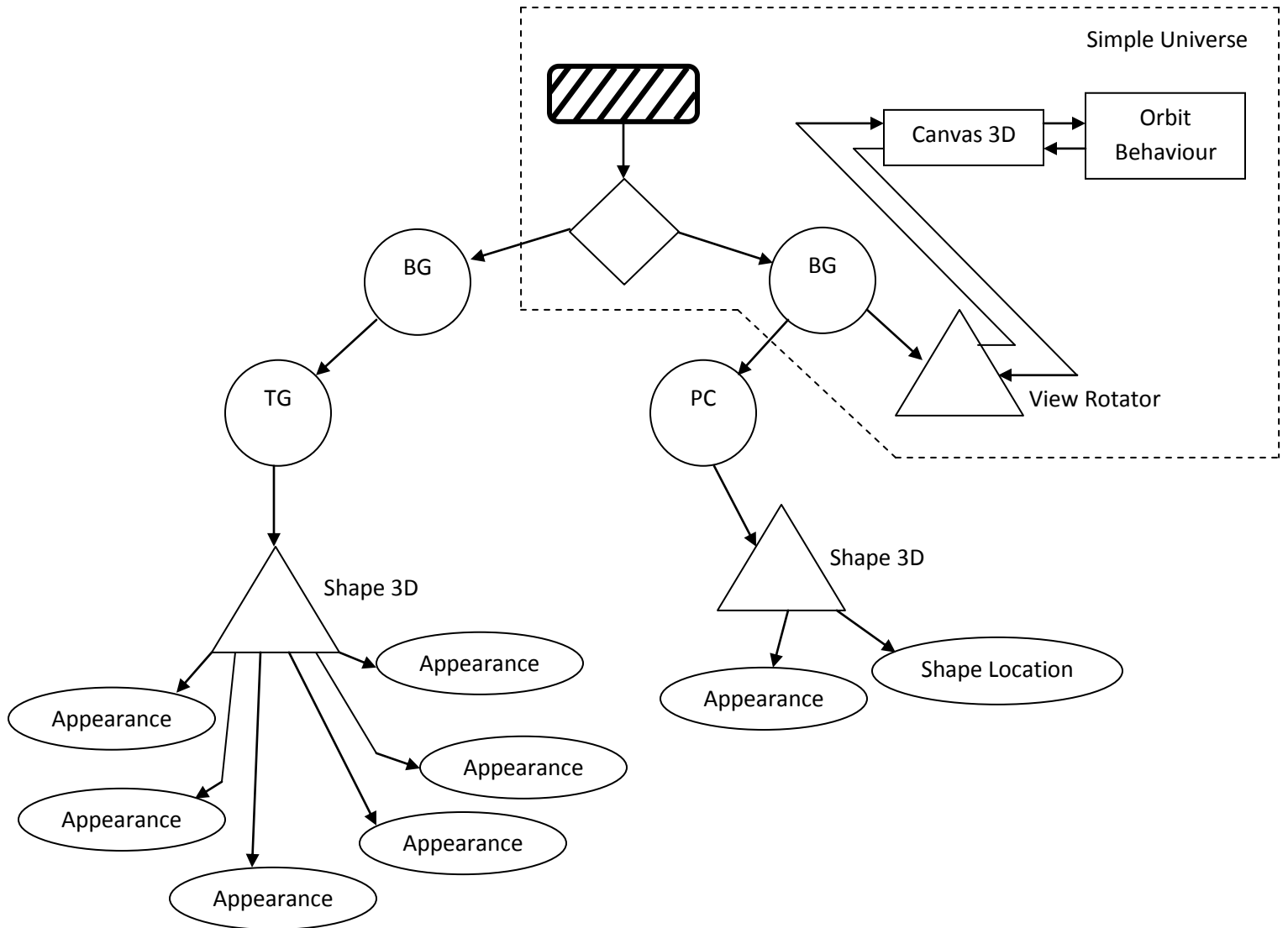
## **Discussion of the Highs and Lows of scene-graph approach (Java 3D) opposed to the state-machine based approach (JOGL)**

The scene-graph approach allowed for a much easier installation (a simple .exe file) as opposed to the state-machine approach which was a bit of a long way round. The scene-graph approach also allows the use of a great mouse interaction with the scene as well as still supporting keyboard, button and checkbox features.

Java 3D also seems to use more memory when run. When two or more Java 3D applets are run on my Windows XP PC; the speed of the pc slows and it takes longer to open each one as opposed to JOGL where about 6 or more JOGL programs were opened and there wasn't much of a problem with processing speed.

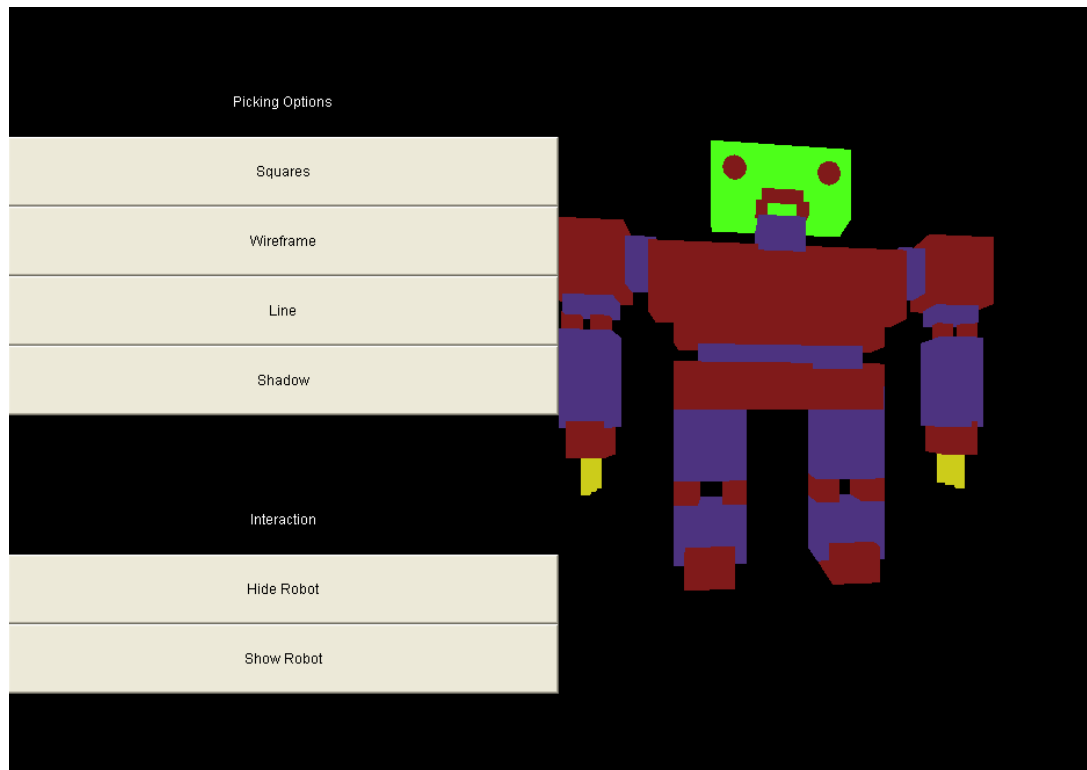
Personally the scene-graph approach felt that it had a greater level of depth as opposed to the state-machine approach (even though they are probably equal). It felt like that and felt less restricting compared to the state machine approach and that allowed for a more positive attitude towards coding various features in and looking into new objects such as Rotation Interpolators.

## Appendix A – Scene Graph of the Code

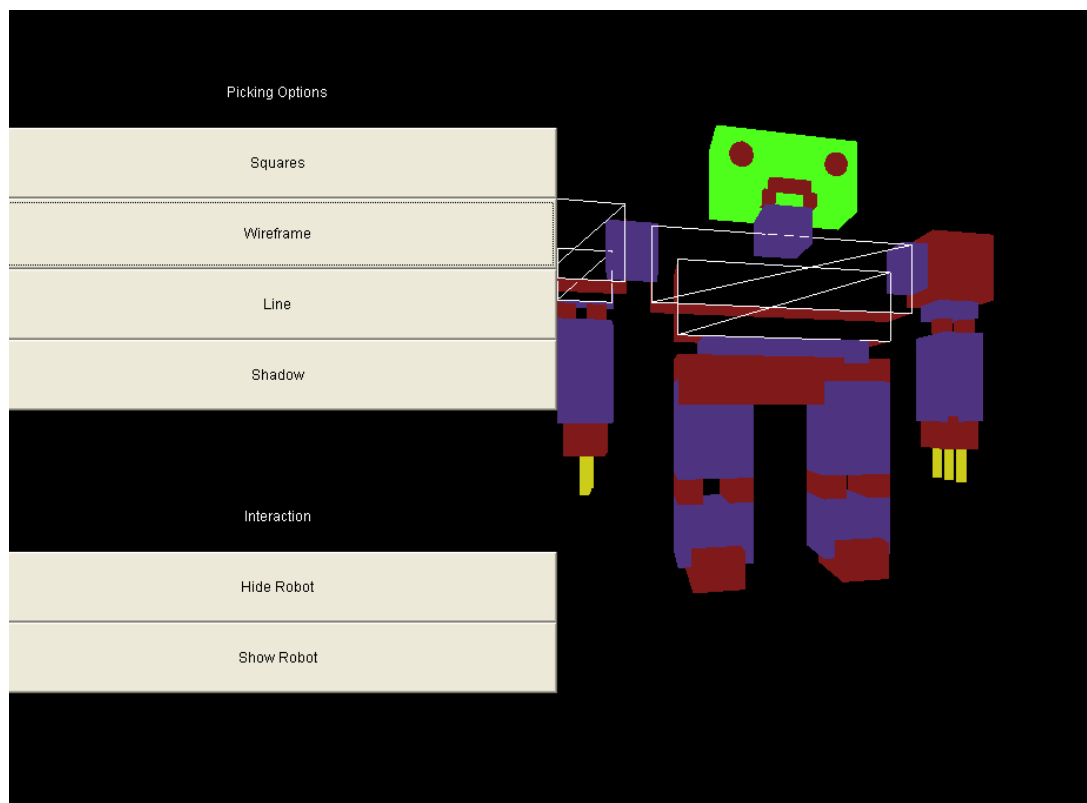


## Appendix B – Screenshots of the Scene

Default View



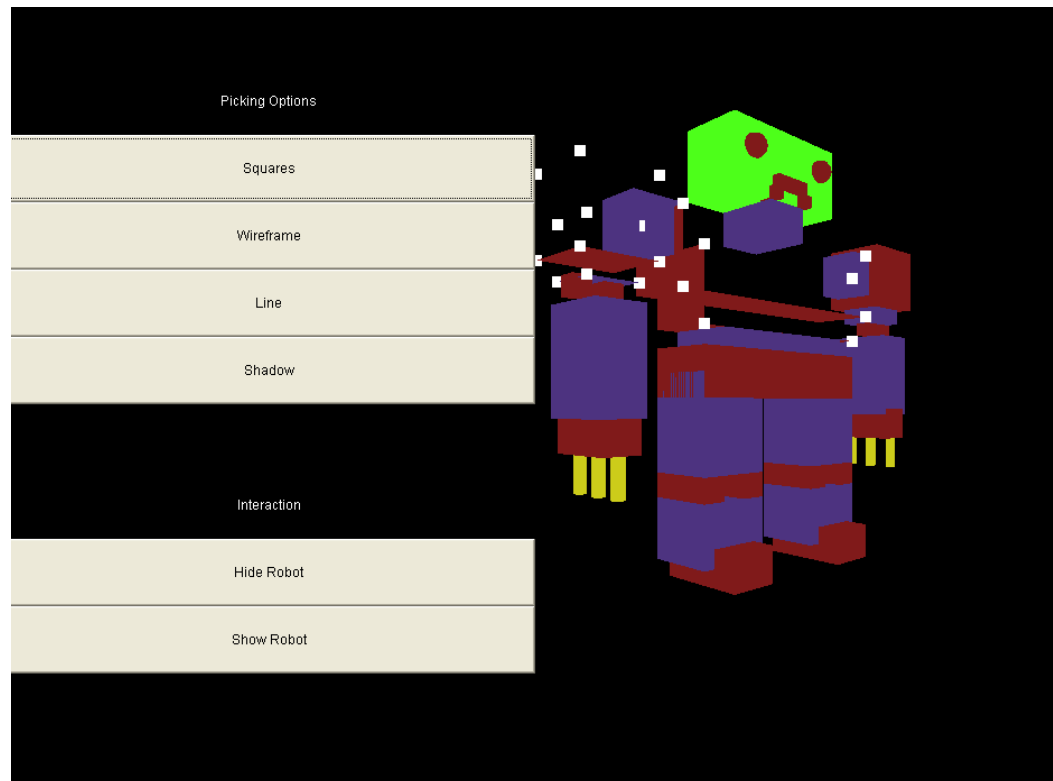
Picking Option  
Wireframe



Picking Option  
Line

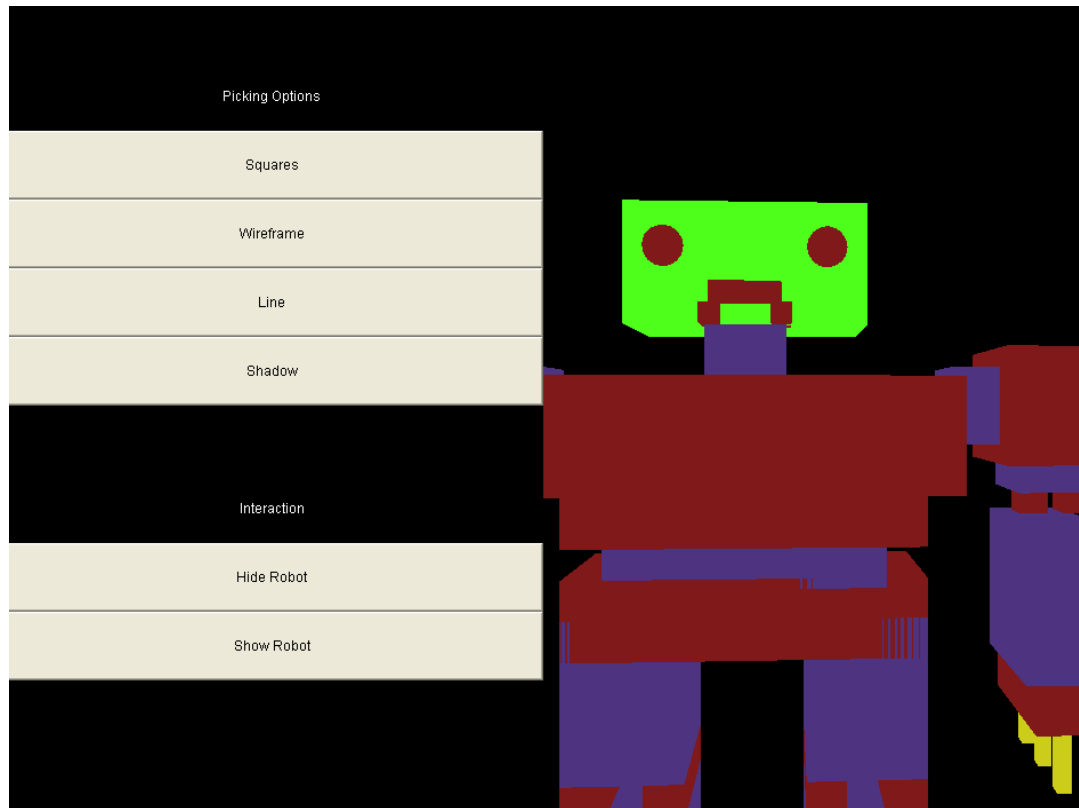


Picking Option  
Squares

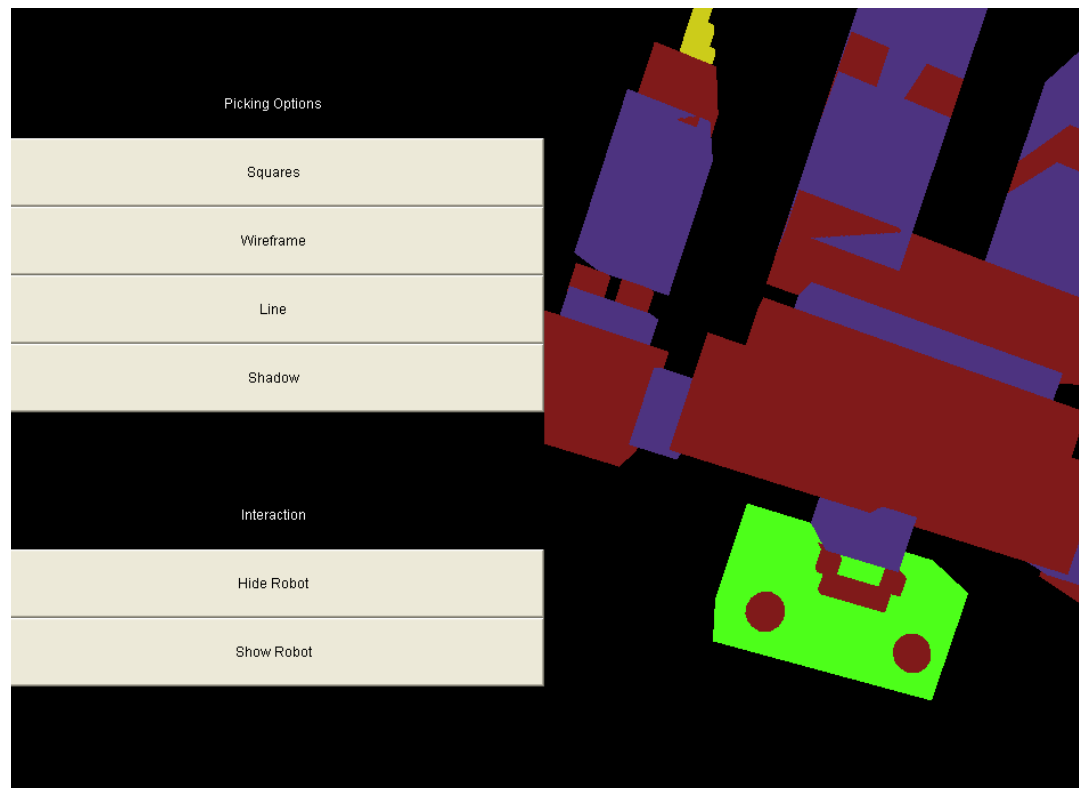




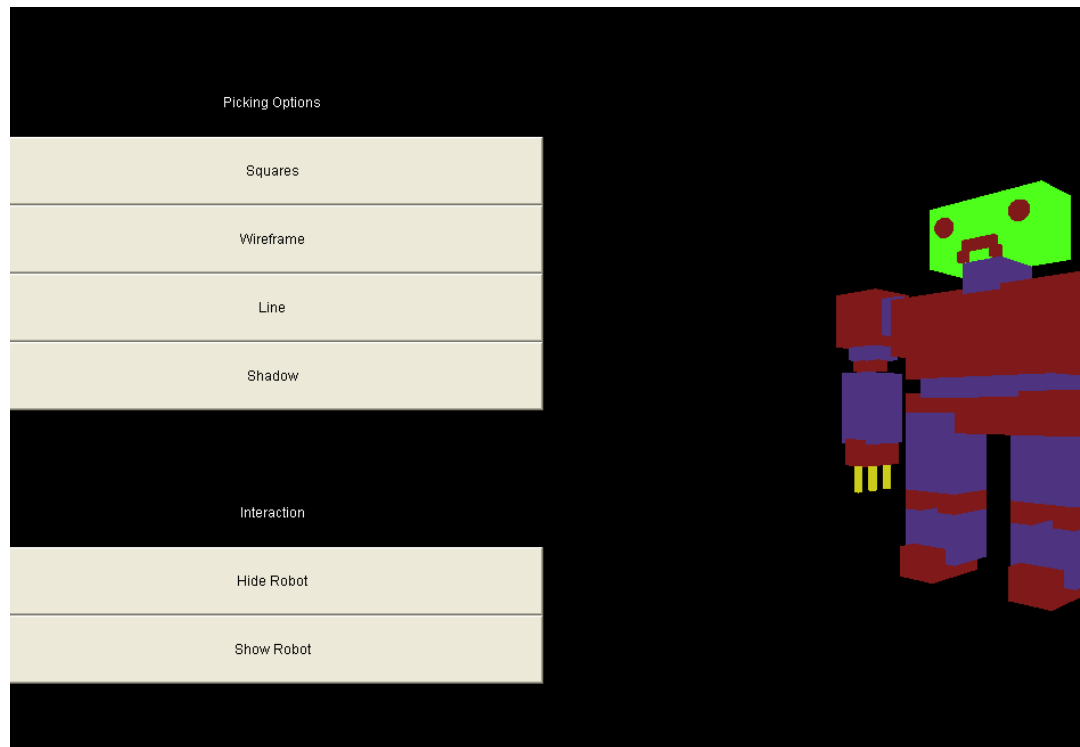
## Mouse Controls Zoom



## Mouse Controls Rotation



Mouse  
Controls  
Translation



## Appendix C – Source Code for the Robot

```
package coursework2_robot;

import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import javax.media.j3d.*;
import javax.vecmath.*;
import com.sun.j3d.utils.applet.MainFrame;
import com.sun.j3d.utils.behaviors.mouse.*;
import com.sun.j3d.utils.behaviors.vp.*;
import com.sun.j3d.utils.geometry.*;
import com.sun.j3d.utils.picking.*;
import com.sun.j3d.utils.universe.*;

public class robot extends Applet implements ActionListener, MouseListener {

    Appearance appear;
    //Appearance wireframe;
    Appearance solidViolet;
    Appearance solidRed;
    Appearance solidGreen;
    Appearance solidYellow;
    AmbientLight ambience;
    SpotLight spotLeft;
    SpotLight spotRight;
    Transform3D trObjects;
```

```
PickCanvas pc;

Canvas3D cv;

BranchGroup bg;

Leaf viewRotator;

SimpleUniverse su;

boolean stopped = false;

int buttonPressAmbient = 0;

int buttonPressEvent = 0;

public static void main(String[] args) {
    new MainFrame(new robot(), 950, 720);
}

@Override

public void init() {
    // create canvas

    setLayout(new BorderLayout());

    Panel p = new Panel();

    p.setLayout(new GridLayout(12, 1));

    p.setBackground(Color.BLACK);

    add(p, BorderLayout.WEST);

    p.add(new Panel());

    Label label = new Label("Picking Options");

    label.setForeground(Color.WHITE);

    label.setAlignment(Label.CENTER);
```

```
p.add(label);

Button button = new Button("Squares");

p.add(button);

button.addActionListener(this);

button = new Button("Wireframe");

p.add(button);

button.addActionListener(this);

button = new Button("Line");

p.add(button);

button.addActionListener(this);

button = new Button("Shadow");

p.add(button);

button.addActionListener(this);


p.add(new Panel());

label = new Label("Interaction");

label.setForeground(Color.WHITE);

label.setAlignment(Label.CENTER);

p.add(label);

button = new Button("Hide Robot");

p.add(button);

button.addActionListener(this);

button = new Button("Show Robot");

p.add(button);

button.addActionListener(this);


GraphicsConfiguration gc =
SimpleUniverse.getPreferredConfiguration();
```

```
/*Canvas3D*/ cv = new Canvas3D(gc);

cv.setBackground(Color.BLACK);

setLayout(new GridLayout(1, 2));

/*SimpleUniverse*/ su = new SimpleUniverse(cv, 2);

su.getViewingPlatform().setNominalViewingTransform();

bg = createSceneGraph();

bg.setCapability(BranchGroup.ALLOW_CHILDREN_READ);

bg.setCapability(BranchGroup.ALLOW_CHILDREN_WRITE);

bg.setCapability(BranchGroup.ALLOW_DETACH);

viewRotator =
createViewRotator(su.getViewingPlatform().getMultiTransformGroup().getTransformGroup(0));

bg.addChild(viewRotator);

//bg.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);

bg.compile();

su.addBranchGraph(bg);

pc = new PickCanvas(cv, bg);

pc.setMode(PickTool.GEOMETRY);

//setLayout(new BorderLayout());

add(cv, BorderLayout.CENTER);

//BranchGroup bg = createSceneGraph();

//bg.compile();

//mouse controls
```

```
OrbitBehavior orbit = new OrbitBehavior(cv);

orbit.setSchedulingBounds(new BoundingSphere());

su.getViewingPlatform().setViewPlatformBehavior(orbit);


cv.addMouseListener(new MouseAdapter() {

    @Override

    public void mouseClicked(MouseEvent ev) {

        System.out.println("no mouse?");


        pc.setShapeLocation(ev);

        //pick all

        PickResult[] results = pc.pickAll();

        if (results == null) {

            return;

        }

        for (int i = 0; i < results.length; i++) {

            Node node = results[i].getObject();

            if (node instanceof Shape3D) {

                ((Shape3D) node).setAppearance(appear);

                System.out.println(node.toString());

            }

        }

    }

})


@Override

public void mousePressed(MouseEvent e) {
```

```
        throw new UnsupportedOperationException("Not supported
yet.");
    }

    @Override
    public void mouseReleased(MouseEvent e) {
        throw new UnsupportedOperationException("Not supported
yet.");
    }
});

System.out.println("The Buttons for the Robot are as follows .. \n"
    + "Mouse Controls: \n"
    + "Left Click (when touching a robot part) - Allows Picking
\n"
    + "Left Click (when not touching robot) - Rotation \n"
    + "Right Click - Translation \n "
    + "Mouse Wheel (Movement) - Zoom In and Out \n"
    + "Mouse Wheel (Click + Drag Mouse) - Zoom In and Out");
}

private BranchGroup createSceneGraph() {
    BranchGroup root = new BranchGroup();
    TransformGroup spin = new TransformGroup();
    spin.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);
    root.addChild(spin);

    //primitives
    appear = new Appearance();
```



```
appear.setMaterial(new Material());

//Box box = new Box(1.2f, 0.3f, 0.8f, ap);#

Box box = new Box();

appear = new Appearance();

appear.setCapability(Appearance.ALLOW_COLORING_ATTRIBUTES_WRITE);
appear.setCapability(Appearance.ALLOW_POINT_ATTRIBUTES_WRITE);
appear.setCapability(Appearance.ALLOW_LINE_ATTRIBUTES_WRITE);
appear.setCapability(Appearance.ALLOW_POLYGON_ATTRIBUTES_WRITE);
appear.setCapability(Appearance.ALLOW_RENDERING_ATTRIBUTES_WRITE);
appear.setCapability(Appearance.ALLOW_MATERIAL_WRITE);

Appearance wireframe = new Appearance();

wireframe.setPolygonAttributes(new PolygonAttributes(
    PolygonAttributes.POLYGON_LINE, PolygonAttributes.CULL_BACK,
0f));

wireframe.setColoringAttributes(new ColoringAttributes(
    0f, 0f, 0f, ColoringAttributes.SHADE_FLAT));

solidViolet = new Appearance();

solidViolet.setPolygonAttributes(new PolygonAttributes(
    PolygonAttributes.POLYGON_FILL, PolygonAttributes.CULL_BACK,
0f));

solidViolet.setColoringAttributes(new ColoringAttributes(
    0.3f, 0.2f, 0.5f, ColoringAttributes.SHADE_GOURAUD));

solidRed = new Appearance();
```

```
solidRed.setPolygonAttributes(new PolygonAttributes(  
    PolygonAttributes.POLYGON_FILL, PolygonAttributes.CULL_BACK,  
0f));  
  
solidRed.setColoringAttributes(new ColoringAttributes(  
    0.5f, 0.1f, 0.1f, ColoringAttributes.SHADE_GOURAUD));  
  
solidGreen = new Appearance();  
solidGreen.setPolygonAttributes(new PolygonAttributes(  
    PolygonAttributes.POLYGON_FILL, PolygonAttributes.CULL_BACK,  
0f));  
  
solidGreen.setColoringAttributes(new ColoringAttributes(  
    0.3f, 1.0f, 0.1f, ColoringAttributes.SHADE_GOURAUD));  
  
solidYellow = new Appearance();  
solidYellow.setPolygonAttributes(new PolygonAttributes(  
    PolygonAttributes.POLYGON_FILL, PolygonAttributes.CULL_BACK,  
0f));  
  
solidYellow.setColoringAttributes(new ColoringAttributes(  
    0.8f, 0.8f, 0.1f, ColoringAttributes.SHADE_GOURAUD));  
  
//rotation of the scene (translation)  
/*Transform3D*/ trObjects = new Transform3D();  
trObjects.setScale(0.2);  
trObjects.setTranslation(new Vector3f(-0.2f, /*-0.2f*/ 0.0f, 0.0f));  
  
TransformGroup tgObjects = new TransformGroup(trObjects);  
spin.addChild(tgObjects);
```

```
TransformGroup tgMouse = tgObjects;

tgMouse.setCapability(TransformGroup.ALLOW_TRANSFORM_WRITE);

tgMouse.setCapability(TransformGroup.ALLOW_TRANSFORM_READ);

//mouse rotation

MouseRotate rotator = new MouseRotate(tgMouse);

BoundingSphere bounds = new BoundingSphere();

rotator.setSchedulingBounds(bounds);

root.addChild(rotator);

//mouse translation

MouseTranslate translator = new MouseTranslate(tgMouse);

translator.setSchedulingBounds(bounds);

root.addChild(translator);

//mouse zoom

MouseZoom zoom = new MouseZoom(tgMouse);

zoom.setSchedulingBounds(bounds);

root.addChild(zoom);


//background and light

Background background= new Background(0.0f, 0.0f, 0.0f);

//background.setColor(0f, 0f, 0f);

//BoundingSphere bounds = new BoundingSphere();

//bounds.setRadius(2);

background.setApplicationBounds(bounds);

root.addChild(background);
```

```
    ambience = new AmbientLight(true, new Color3f(Color.green));
    ambience.setInfluencingBounds(bounds);
    ambience.setCapability(Light.ALLOW_STATE_WRITE);
    root.addChild(ambience);

    spotLeft = new SpotLight(new Color3f(Color.orange), new Point3f(0.7f,
0.7f, 0.7f),
        new Point3f(1f, 0f, 0f), new Vector3f(-0.7f, -0.7f, -0.7f),
(float) (Math.PI / 6.0), 0f);
    spotLeft.setCapability(Light.ALLOW_STATE_WRITE);
    spotLeft.setInfluencingBounds(bounds);
    //root.addChild(spotLeft);

    spotRight = new SpotLight(new Color3f(Color.orange), new
Point3f(1.0f, -0.7f, 0.0f),
        new Point3f(1f, 0f, 0f), new Vector3f(0.7f, 0.7f, 0.0f),
(float) (Math.PI / 12.0), 128f);
    spotRight.setCapability(Light.ALLOW_STATE_WRITE);
    spotRight.setInfluencingBounds(bounds);
    //root.addChild(spotRight);

    //upper arms
    Transform3D trBox = new Transform3D();
    trBox.setTranslation(new Vector3f(-3.2f, 2.6f, 0.0f));
    box = new Box(0.6f, 0.6f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);
    TransformGroup tgBox = new TransformGroup(trBox);
```

```
tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(3.2f, 2.6f, 0.0f));

box = new Box(0.6f, 0.6f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-2.5f, 2.6f, 0.0f));

box = new Box(0.4f, 0.4f, 0.4f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(2.5f, 2.6f, 0.0f));

box = new Box(0.4f, 0.4f, 0.4f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);
```

```
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-3.2f, 2.1f, 0.0f));
box = new Box(0.4f, 0.4f, 0.4f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidViolet);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(3.2f, 2.1f, 0.0f));
box = new Box(0.4f, 0.4f, 0.4f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidViolet);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-3.52f, 1.6f, 0.0f));
box = new Box(0.15f, 0.15f, 0.15f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
```

```
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-3.05f, 1.6f, 0.0f));
box = new Box(0.15f, 0.15f, 0.15f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(3.05f, 1.6f, 0.0f));
box = new Box(0.15f, 0.15f, 0.15f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(3.52f, 1.6f, 0.0f));
box = new Box(0.15f, 0.15f, 0.15f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
```

```
tgBox.addChild(box);

//lower arms

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-3.2f, 0.6f, 0.0f));
box = new Box(0.4f, 0.8f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidViolet);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(3.2f, 0.6f, 0.0f));
box = new Box(0.4f, 0.8f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidViolet);
tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);

//hands + fingers

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-3.2f, -0.4f, 0.0f));
```



```
box = new Box(0.3f, 0.3f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(3.2f, -0.4f, 0.0f));

box = new Box(0.3f, 0.3f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


//left

trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-3.2f, -0.8f, 0.5f));

box = new Box(0.07f, 0.5f, 0.07f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidYellow);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();
```

```
trBox.setTranslation(new Vector3f(-3.2f, -0.8f, 0.0f));

box = new Box(0.07f, 0.5f, 0.07f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidYellow);

tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);


trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-3.2f, -0.8f, -0.5f));
box = new Box(0.07f, 0.5f, 0.07f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidYellow);

tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);


//right

trBox = new Transform3D();
trBox.setTranslation(new Vector3f(3.2f, -0.8f, 0.5f));
box = new Box(0.07f, 0.5f, 0.07f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidYellow);

tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);
```

```
trBox = new Transform3D();

trBox.setTranslation(new Vector3f(3.2f, -0.8f, 0.0f));

box = new Box(0.07f, 0.5f, 0.07f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidYellow);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);
```

```
trBox = new Transform3D();

trBox.setTranslation(new Vector3f(3.2f, -0.8f, -0.5f));

box = new Box(0.07f, 0.5f, 0.07f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidYellow);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);
```

```
//upper legs
```

```
trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 0.5f, 0.0f));

box = new Box(1.8f, 0.4f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
```

```
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(1.2f, -0.3f, 0.0f));

box = new Box(0.6f, 0.8f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-1.2f, -0.3f, 0.0f));

box = new Box(0.6f, 0.8f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-1.6f, -1.3f, 0.0f));

box = new Box(0.2f, 0.2f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
```

```
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-0.8f, -1.3f, 0.0f));

box = new Box(0.2f, 0.2f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.8f, -1.3f, 0.0f));

box = new Box(0.2f, 0.2f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(1.6f, -1.3f, 0.0f));

box = new Box(0.2f, 0.2f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
```

```
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


//lower legs


trBox = new Transform3D();
trBox.setTranslation(new Vector3f(1.2f, -2.0f, 0.0f));
box = new Box(0.6f, 0.5f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);


trBox = new Transform3D();
trBox.setTranslation(new Vector3f(-1.2f, -2.0f, 0.0f));
box = new Box(0.6f, 0.5f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);
tgObjects.addChild(tgBox);
tgBox.addChild(box);
```

```
trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-1.2f, -2.5f, 0.25f));

box = new Box(0.4f, 0.3f, 0.8f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(1.2f, -2.5f, 0.25f));

box = new Box(0.4f, 0.3f, 0.8f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
    | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


//body


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 2.3f, 0.0f));

box = new Box(2.2f, 0.6f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
    | Primitive.ENABLE_APPEARANCE_MODIFY
```

```
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 1.8f, 0.0f));

box = new Box(1.8f, 0.6f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 1.0f, 0.0f));

box = new Box(1.4f, 0.2f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidViolet);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


//head

trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 4.0f, 0.0f));

box = new Box(1.2f, 0.6f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
```



```
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidGreen);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


Transform3D trSphere = new Transform3D();

trSphere.setTranslation(new Vector3f(0.8f, 4.15f, 0.65f));

Sphere sphere = new Sphere(0.2f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

TransformGroup tgSphere = new TransformGroup(trSphere);

tgObjects.addChild(tgSphere);

tgSphere.addChild(sphere);


trSphere = new Transform3D();

trSphere.setTranslation(new Vector3f(-0.8f, 4.15f, 0.65f));

sphere = new Sphere(0.2f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgSphere = new TransformGroup(trSphere);

tgObjects.addChild(tgSphere);

tgSphere.addChild(sphere);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 3.7f, 0.55f));

box = new Box(0.35f, 0.1f, 0.2f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
```

```
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.35f, 3.5f, 0.55f));

box = new Box(0.10f, 0.1f, 0.2f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(-0.35f, 3.5f, 0.55f));

box = new Box(0.10f, 0.1f, 0.2f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
        | Primitive.GENERATE_NORMALS, solidRed);

tgBox = new TransformGroup(trBox);

tgObjects.addChild(tgBox);

tgBox.addChild(box);


trBox = new Transform3D();

trBox.setTranslation(new Vector3f(0.0f, 3.10f, 0.0f));

box = new Box(0.4f, 0.3f, 0.6f, Primitive.ENABLE_GEOMETRY_PICKING
        | Primitive.ENABLE_APPEARANCE_MODIFY
```

```
        | Primitive.GENERATE_NORMALS, solidViolet);

    tgBox = new TransformGroup(trBox);

    tgObjects.addChild(tgBox);

    tgBox.addChild(box);

    return root;
}

public void mouseClicked(MouseEvent mouseEvent) {

}

public void mouseEntered(MouseEvent mouseEvent) {

}

public void mouseExited(MouseEvent mouseEvent) {

}

public void actionPerformed(ActionEvent actionEvent) {

    String ac = actionEvent.getActionCommand();

    if (ac.equals("Squares")) {

        System.out.println("sdgfhjkl");

        appear.setPolygonAttributes(new PolygonAttributes(

            PolygonAttributes.POLYGON_POINT,
            PolygonAttributes.CULL_BACK, 0));

        appear.setPointAttributes(new PointAttributes(10, false));

        //BranchGroup createSceneGraph = createSceneGraph();

    } else if (ac.equals("Wireframe")) {
```

```
System.out.println("wireframe");

appear.setPolygonAttributes(new PolygonAttributes(
    PolygonAttributes.POLYGON_LINE,
    PolygonAttributes.CULL_BACK, 0f));

appear.setColoringAttributes(new ColoringAttributes(
    1f, 1f, 1f, ColoringAttributes.SHADE_FLAT));

} else if (ac.equals("Line")) {
    System.out.println("outline");

    ColoringAttributes ca = new ColoringAttributes();
    ca.setColor(1f, 1f, 1f);
    appear.setColoringAttributes(ca);

    appear.setPolygonAttributes(new
    PolygonAttributes(PolygonAttributes.POLYGON_LINE,
        PolygonAttributes.CULL_BACK, 0));

    appear.setLineAttributes(new LineAttributes(6,
        LineAttributes.PATTERN_DASH, false));

} else if (ac.equals("Shadow")) {
    System.out.println("shadow");

    ColoringAttributes ca = new ColoringAttributes();
    ca.setColor(1f, 1f, 1f);

    //appear.setColoringAttributes(ca);

    appear.setPolygonAttributes(new
    PolygonAttributes(PolygonAttributes.POLYGON_FILL,
        PolygonAttributes.CULL_BACK, 0));

    RenderingAttributes ra = new RenderingAttributes();
    ra.setIgnoreVertexColors(true);
```

```
        appear.setRenderingAttributes(ra);

    } else if (ac.equals("Hide Robot")) {

        System.out.println("hide method");

        bg.detach();

    } else if (ac.equals("Show Robot")) {

        System.out.println("show method");

        bg = createSceneGraph();

        bg.setCapability(BranchGroup.ALLOW_CHILDREN_READ);

        bg.setCapability(BranchGroup.ALLOW_CHILDREN_WRITE);

        bg.setCapability(BranchGroup.ALLOW_DETACH);

        viewRotator =
createViewRotator(su.getViewingPlatform().getMultiTransformGroup().getTransformGroup(0));

        bg.addChild(viewRotator);

        bg.compile();

        su.addBranchGraph(bg);

    }

}

private Leaf createViewRotator(TransformGroup tgView) {

    //view rotator

    Alpha alpha = new Alpha(-1, 4000);
```

```
        RotationInterpolator rotator = new RotationInterpolator(alpha,
tgView);

        //RotationInterpolator rotator = new RotationInterpolator(null,
null);

        BoundingSphere bounds = new BoundingSphere();

        rotator.setSchedulingBounds(bounds);

        return rotator;
    }

    @Override
    public void mousePressed(MouseEvent e) {
        throw new UnsupportedOperationException("Not supported yet.");
    }

    @Override
    public void mouseReleased(MouseEvent e) {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```