**海南大学计算机科学与技术学院**

**《计算机图形学》课内实验报告三**

****

**班 级：计算机科学与技术2021-3班**

**成 员： 李季鸿 20213002624**

**报告名称： 实验报告三**

**指导老师： 高新瑞**

**完成日期： 2023年9月28日**

**《计算机图形学》实验报告三**

**学生姓名：李季鸿 班级：21级计科3班 学号：20213002624**

**实验地点：9-202 指导教师：高新瑞**

**实验日期：**2023.9.28  **实验课时：2学时**

**实验环境：**Windows 10+JDK1.8+记事本+IntelliJ IDEA

**一、实验目的**

**实验一目的：**

通过编写Java 3D程序，展示如何创建和显示彩色线条，以及如何使用鼠标进行交互操作，包括旋转、平移和缩放。这个程序创建了一个简单的Java 3D场景，其中包含四条彩色线条，并允许用户使用鼠标进行互动。以下是实验的主要目的：

1. 创建一个Java 3D应用程序，用于显示三维图形。

2. 在Java 3D场景中创建一个带有彩色线条的3D对象。

3. 定义线条的顶点坐标和颜色，以在场景中呈现彩色线条。

4. 配置线条的属性，如线宽度和线型。

5. 添加交互性，使用户能够使用鼠标旋转、平移和缩放整个场景。

6. 显示背景颜色，以提高场景的可视效果。

7. 编写一个主类，通过MainFrame启动Java 3D应用程序。

**实验二目的：**

本实验旨在演示如何使用Java 3D库中的LineStripArray类创建和显示带有不同分段的彩色线条，并且通过鼠标交互实现视图的旋转、平移和缩放。以下是实验的主要目的：

1. Java 3D应用程序：创建一个基于Java 3D库的应用程序，用于显示和操作三维图形。

2. 线条创建：使用LineStripArray类创建包含彩色线条的3D对象。本实验中的线条由多个顶点组成，每个顶点都具有坐标和颜色。

3. 线条分段：演示了不同的线条分段方案，其中线条分为不同的子组，以便展示不同的效果。

- 方案一：4个顶点、4个顶点

- 方案二：2个顶点、6个顶点

- 方案三：6个顶点、2个顶点

- 方案四：整个线条作为一个组

4. 线条属性：配置线条的属性，包括线宽度、线型和线条的抗锯齿特性。

5. 鼠标交互：通过MouseRotate、MouseZoom和MouseTranslate等鼠标行为，使用户能够与场景进行交互，改变视图的角度和位置。

6. 背景颜色：设置场景的背景颜色，以提高可视效果。

7. 主类：编写一个主类，通过MainFrame启动Java 3D应用程序，使其可以在窗口中运行。

通过这个实验，可以学到如何使用Java 3D库中的LineStripArray类创建具有不同分段的彩色线条，并实现鼠标交互，以便更好地理解和掌握Java 3D的基本概念和功能。这个实验还可以作为进一步学习和开发更复杂Java 3D应用程序的基础。

**二、实验过程**

## 实验内容一：

1. 代码
2. */\*\**
3. \* \\* Created with IntelliJ IDEA.
4. \* \\* @ProjectName: 例3.4 线显示的程序
5. \* \\* @FileName: DisplayLines
6. \* \\* @author: li-jihong
7. \* \\* Date: 2023-09-28 10:41
8. \*/
9. import com.sun.j3d.utils.applet.MainFrame;
10. import com.sun.j3d.utils.behaviors.mouse.MouseRotate;
11. import com.sun.j3d.utils.behaviors.mouse.MouseTranslate;
12. import com.sun.j3d.utils.behaviors.mouse.MouseZoom;
13. import com.sun.j3d.utils.universe.SimpleUniverse;
14. import javax.media.j3d.\*;
15. import javax.vecmath.Color3f;
16. import javax.vecmath.Point3d;
17. import javax.vecmath.Vector3f;
18. import java.applet.Applet;
19. import java.awt.\*;
20. public class DisplayLines extends Applet {
21. public DisplayLines() {
22. setLayout(new BorderLayout());
23. GraphicsConfiguration gc = SimpleUniverse.getPreferredConfiguration();
24. Canvas3D c = new Canvas3D(gc);
25. add("Center", c);
26. BranchGroup BranchGroupScene = createBranchGroupSceneGraph();
27. SimpleUniverse u = new SimpleUniverse(c);
28. u.getViewingPlatform().setNominalViewingTransform();
29. u.addBranchGraph(BranchGroupScene);
30. }
31. public static void main(String[] args) {
32. new MainFrame(new DisplayLines(), 450, 450);
33. }
34. public BranchGroup createBranchGroupSceneGraph() {
35. BranchGroup BranchGroupRoot = new BranchGroup();
36. BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), 100.0);
37. Color3f bgColor = new Color3f(1.0f, 1.0f, 1.0f);
38. Background bg = new Background(bgColor);
39. bg.setApplicationBounds(bounds);
40. BranchGroupRoot.addChild(bg);
41. Color3f directionalColor = new Color3f(1.f, 0.f, 0.f);
42. Vector3f vec = new Vector3f(0.f, 0.f, -1.0f);
43. DirectionalLight directionalLight = new DirectionalLight(directionalColor, vec);
44. directionalLight.setInfluencingBounds(bounds);
45. BranchGroupRoot.addChild(directionalLight);
46. TransformGroup transformgroup = new TransformGroup();
47. transformgroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);
48. transformgroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_READ);
49. BranchGroupRoot.addChild(transformgroup);
50. MouseRotate mouserotate = new MouseRotate();
51. mouserotate.setTransformGroup(transformgroup);
52. BranchGroupRoot.addChild(mouserotate);
53. mouserotate.setSchedulingBounds(bounds);
54. MouseZoom mousezoom = new MouseZoom();
55. mousezoom.setTransformGroup(transformgroup);
56. BranchGroupRoot.addChild(mousezoom);
57. mousezoom.setSchedulingBounds(bounds);
58. MouseTranslate mousetranslate = new MouseTranslate();
59. mousetranslate.setTransformGroup(transformgroup);
60. BranchGroupRoot.addChild(mousetranslate);
61. mousetranslate.setSchedulingBounds(bounds);
62. transformgroup.addChild(shapelines());*//将线加入到TransformGroup中*
63. BranchGroupRoot.compile();
64. return BranchGroupRoot;
65. }
66. public Shape3D shapelines() {
67. Shape3D shapelines0 = new Shape3D();
68. *//定义点坐标*
69. float vertexes[] = {.8f, 0.5f, 0.0f, -0.8f, .8f, 0.0f,
70. 0.8f, 0.0f, 0.0f, -0.8f, 0.5f, 0.0f,
71. 0.8f, -0.5f, 0.0f, -0.8f, 0.2f, 0.0f};
72. *//定义颜色*
73. float colors[] = {1.0f, 0.f, .0f, 0.0f, 1.f, .0f,
74. 0.0f, 0.f, 1.f, 1.0f, 1.0f, 0.f,
75. 0.0f, 1.0f, 1.f, 1.f, 0.f, 1.0f};
76. *//定义线数组*
77. LineArray lines = new LineArray(6, LineArray.COORDINATES | LineArray.COLOR\_3);
78. lines.setCoordinates(0, vertexes);
79. lines.setColors(0, colors);
80. *//定义线属性*
81. LineAttributes lineattributes = new LineAttributes();
82. lineattributes.setLineWidth(20.f);
83. lineattributes.setLineAntialiasingEnable(false);
84. lineattributes.setLinePattern(LineAttributes.PATTERN\_SOLID);
85. *//PATTERN\_SOLID PATTERN\_DASH PATTERN\_DOT PATTERN\_DASH\_DOT*
86. Appearance app = new Appearance();
87. app.setLineAttributes(lineattributes);
88. shapelines0.setGeometry(lines);
89. shapelines0.setAppearance(app);
90. return shapelines0;
91. }
92. }

（2）结果截图

|  |  |
| --- | --- |
|  |  |
| 图3-1 使用反走样参数 | 图3-2 未使用反走样参数 |

## 实验内容二：

（1）代码

1. */\*\**
2. \* \\* Created with IntelliJ IDEA.
3. \* \\* @ProjectName: 例3.6 LineStripArray类程序实例
4. \* \\* @FileName: DisplayLineStrip
5. \* \\* @author: li-jihong
6. \* \\* Date: 2023-09-21 14:12
7. \*/
8. import com.sun.j3d.utils.applet.MainFrame;
9. import com.sun.j3d.utils.behaviors.mouse.MouseRotate;
10. import com.sun.j3d.utils.behaviors.mouse.MouseTranslate;
11. import com.sun.j3d.utils.behaviors.mouse.MouseZoom;
12. import com.sun.j3d.utils.universe.SimpleUniverse;
13. import javax.media.j3d.\*;
14. import javax.vecmath.Color3f;
15. import javax.vecmath.Point3d;
16. import javax.vecmath.Vector3f;
17. import java.applet.Applet;
18. import java.awt.\*;
19. public class DisplayLineStrip extends Applet {
20. public DisplayLineStrip() {
21. setLayout(new BorderLayout());
22. GraphicsConfiguration gc = SimpleUniverse.getPreferredConfiguration();
23. Canvas3D c = new Canvas3D(gc);
24. add("Center", c);
25. BranchGroup BranchGroupScene = createBranchGroupSceneGraph();
26. SimpleUniverse u = new SimpleUniverse(c);
27. u.getViewingPlatform().setNominalViewingTransform();
28. u.addBranchGraph(BranchGroupScene);
29. }
30. public static void main(String[] args) {
31. new MainFrame(new DisplayLineStrip(), 450, 450);
32. }
33. public BranchGroup createBranchGroupSceneGraph() {
34. BranchGroup BranchGroupRoot = new BranchGroup();
35. BoundingSphere bounds = new BoundingSphere(new Point3d(0.0, 0.0, 0.0), 100.0);
36. Color3f bgColor = new Color3f(1.0f, 1.0f, 1.0f);
37. Background bg = new Background(bgColor);
38. bg.setApplicationBounds(bounds);
39. BranchGroupRoot.addChild(bg);
40. Color3f directionalColor = new Color3f(1.f, 0.f, 0.f);
41. Vector3f vec = new Vector3f(0.f, 0.f, -1.0f);
42. DirectionalLight directionalLight = new DirectionalLight(directionalColor, vec);
43. directionalLight.setInfluencingBounds(bounds);
44. BranchGroupRoot.addChild(directionalLight);
45. TransformGroup transformgroup = new TransformGroup();
46. transformgroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_WRITE);
47. transformgroup.setCapability(TransformGroup.ALLOW\_TRANSFORM\_READ);
48. BranchGroupRoot.addChild(transformgroup);
49. MouseRotate mouserotate = new MouseRotate();
50. mouserotate.setTransformGroup(transformgroup);
51. BranchGroupRoot.addChild(mouserotate);
52. mouserotate.setSchedulingBounds(bounds);
53. MouseZoom mousezoom = new MouseZoom();
54. mousezoom.setTransformGroup(transformgroup);
55. BranchGroupRoot.addChild(mousezoom);
56. mousezoom.setSchedulingBounds(bounds);
57. MouseTranslate mousetranslate = new MouseTranslate();
58. mousetranslate.setTransformGroup(transformgroup);
59. BranchGroupRoot.addChild(mousetranslate);
60. mousetranslate.setSchedulingBounds(bounds);
61. transformgroup.addChild(Striplines());
62. BranchGroupRoot.compile();
63. return BranchGroupRoot;
64. }
65. public Shape3D Striplines() {
66. Shape3D Striplines0 = new Shape3D();
67. float vertexes[] = {-0.8f, 0.5f, 0.0f, 0.8f, 0.9f, 0.0f,
68. -0.8f, 0.2f, 0.0f, 0.8f, 0.7f, 0.0f,
69. -0.8f, -0.2f, 0.0f, 0.8f, -0.5f, 0.0f,
70. -0.8f, -0.5f, 0.0f, 0.8f, -.8f, 0.0f,};
71. float colors[] = {1.0f, 0.f, .0f, 0.0f, 1.f, .0f,
72. 0.0f, 0.f, 1.f, 1.0f, 1.0f, 0.f,
73. 0.0f, 1.0f, 1.f, 1.f, 0.f, 1.0f,
74. 0.0f, .0f, 0.f, 0.3f, 0.8f, 0.0f,};
75. *//方案一: 分为两组，4个顶点、4个顶点*
76. *//        int[] substrips = new int[2];*
77. *//        substrips[0] = 4;*
78. *//        substrips[1] = 4;*
79. *//方案二: 分为两组，2个顶点、6个顶点*
80. *//        int[] substrips = new int[2];*
81. *//        substrips[0] = 2;*
82. *//        substrips[1] = 6;*
83. *//方案三: 分为两组，6个顶点、2个顶点*
84. *//        int[] substrips = new int[2];*
85. *//        substrips[0] = 6;*
86. *//        substrips[1] = 2;*
87. *//方案四: 整个是一个组*
88. int[] substrips=new int[1];
89. substrips[0]=8;
90. LineStripArray Striplines = new LineStripArray
91. (8, LineArray.COORDINATES | LineArray.COLOR\_3, substrips);
92. Striplines.setCoordinates(0, vertexes);
93. Striplines.setColors(0, colors);
94. LineAttributes lineattributes = new LineAttributes();
95. lineattributes.setLineWidth(9.0f);
96. lineattributes.setLineAntialiasingEnable(true);
97. lineattributes.setLinePattern(0);
98. Appearance app = new Appearance();
99. app.setLineAttributes(lineattributes);
100. Striplines0.setGeometry(Striplines);
101. Striplines0.setAppearance(app);
102. return Striplines0;
103. }
104. }

（2）结果截图

|  |  |
| --- | --- |
|  |  |
| 图3-3分为两组，4个顶点、4个顶点 | 图3-4分为两组，2个顶点、6个顶点 |
|  |  |
| 图3-5分为两组，6个顶点、2个顶点 | 图3-6 整个是一个组 |

**三、实验总结**

**实验一总结：**

在这个实验中，成功地创建了一个简单的Java 3D应用程序，展示了如何创建和显示彩色线条，并实现了鼠标交互操作。以下是实验的主要总结点：

1. Java 3D应用程序：创建了一个基于Java 3D库的应用程序，用于呈现三维图形。

2. 线条的创建：通过定义顶点坐标和颜色，创建了四条彩色线条，并将它们添加到场景中。

3. 线条属性：配置了线条的属性，包括线宽度和线型，以改变线条的外观。

4. 鼠标交互：通过使用Java 3D的鼠标旋转、平移和缩放行为，使用户能够与场景进行交互，改变视图的角度和位置。

5. 背景颜色：设置了场景的背景颜色，以增强可视效果。

6. 主类：编写了一个主类，通过MainFrame启动Java 3D应用程序，使其可以在窗口中运行。

通过这个实验，我学到了如何使用Java 3D库创建基本的三维图形，以及如何实现用户与场景的交互。这是一个入门级的示例，可以作为学习和探索更复杂Java 3D应用程序开发的基础。在实践中，可以进一步扩展这个应用程序，添加更多的三维对象和交互功能，以满足特定需求。

**实验二总结：**

在这个实验中，成功地创建了一个Java 3D应用程序，展示了如何使用LineStripArray类创建和显示带有不同分段的彩色线条，并实现了鼠标交互操作。以下是实验的主要总结点：

1. Java 3D应用程序：创建了一个基于Java 3D库的应用程序，用于呈现三维图形。

2. 线条的创建：通过使用LineStripArray类，创建了一组彩色线条，每个线条由多个顶点组成。还演示了不同的线条分段方案，以展示不同的效果。

3. 线条属性：配置了线条的属性，包括线宽度、线型和抗锯齿特性，以改变线条的外观。

4. 鼠标交互：通过使用Java 3D的鼠标旋转、平移和缩放行为，使用户能够与场景进行交互，改变视图的角度和位置。

5. 背景颜色：设置了场景的背景颜色，以增强可视效果。

6. 主类：编写了一个主类，通过MainFrame启动Java 3D应用程序，使其可以在窗口中运行。

通过这个实验，我学到了如何使用Java 3D库创建具有不同分段的彩色线条，以及如何实现用户与场景的交互。这是一个入门级的示例，可作为学习和探索更复杂Java 3D应用程序开发的基础。进一步的学习可以包括创建更复杂的三维对象，添加纹理和光照效果，以及实现更高级的交互功能，以满足特定需求。通过不断练习和扩展这些概念，可以更深入地理解和应用Java 3D技术。