3D Web Graphics without Plugins using VML

Student: Jiewei Lin
Advisor: Dr. Chris Pollett
Committee Members: Dr. Sin-Min Lee
Dr. Ho Kuen Ng

May 20, 2003
Overview

- Introduction
- Deployment requirements
- Implementation highlights
- Maximum Load
- Limitations
- Conclusion
A VRML plugin is required to view VRML documents in Internet Explorer.

Eg. Cortona
The goal of this project is to develop a stylesheet-transformation from the X3D language to VML.
An X3D document.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D SYSTEM "latest.dtd">  
<X3D>
  <Scene>
    <Shape>
      <Appearance>
        <Material diffuseColor="1.0 1.0 0.0"/>
      </Appearance>
      <Box size="9 15 4.5"/>
    </Shape>
  </Scene>
</X3D>
```
An VML document.

```html
<html xmlns:v="urn:schemas-microsoft-com:vml"
xmlns:o="urn:schemas-microsoft-com:office:office"
xmlns="http://www.w3.org/TR/REC-html40">
  
  <head>
    <style>
      v\:* {behavior:url(#default#VML);}
    </style>
  </head>
  
  <body>
      <v:stroke on="false"/>
      <v:fill method="linear sigma" angle="45" type="gradient" />
    </v:polyline>
  </body>
</html>
```
Introduction (cont.)

An XSL Stylesheet

- X3D file
- x3dToVml.xsl

XSL Processor (IE v6)

VML Doc.
Requirements

1. An X3D input document
2. X3dToVml.dtd
3. X3dToVml.xsl
4. X3dToVml.html
5. Web browser
1. A sample X3D input document

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="X3dToVml.xsl"?>
<!DOCTYPE X3D SYSTEM " X3dToVml.dtd">
<X3D>
  <Scene>
    <Shape>
      <Appearance><Material/></Appearance>
      <Box size="200.0 300.0 400.0"/>
    </Shape>
  </Scene>
</X3D>
```
1a. X3D tags supported.

- X3D
- Scene
- Group
  - DEF
  - USE
- Transform
  - translation "0 0 0"
  - rotation "1 0 0 0"
  - scale "1 1 1"
- Shape
- Appearance
- Material
  - diffuseColor "0 0 1"
- Box
  - size "50 50 50"
- Cone
  - bottomRadius "50"
  - Height "100"
- Cylinders
  - height "100"
  - Radius "50"
- Sphere
  - radius "50"
2. X3dToVml.dtd

(a code fragment from file “X3dToVml.dtd”)

<!ENTITY % PrimitiveNodes "(Box | Cylinder | Cone | Sphere)">

<!ELEMENT Box EMPTY>
<!ATTLIST Box
    size CDATA "50 50 50">

<!ELEMENT Cylinder EMPTY>
<!ATTLIST Cylinder
    height CDATA "100"
    radius CDATA "50"
3a. X3dToVml.xsl: JavaScript section

```xml
<msxsl:script language="JavaScript1.2" implements-prefix="project">
<!CDATA[

    var sceneArray = new Array();

    // a lot of JavaScript code is deleted

    function createBox(width, height, depth)
    {
        var box = new Box(width, height, depth);
        return box.toString();
    }

]]>
</msxsl:script>
```
3b. X3dToVml.xsl: template matching

<!-- calling a JavaScript function -->
<xsl:template match="Box">

<!-- gets box's size from box tag -->
<xsl:variable name="boxDim" select="@size"/>
<xsl:variable name="x" select="substring-before($boxDim, ' ')"/>
<xsl:variable name="rest" select="substring-after($boxDim, ' ')"/>
<xsl:variable name="y" select="substring-before($rest, ' ')"/>
<xsl:variable name="z" select="substring-after($rest, ' ')"/>  
<xsl:variable name="createBox" select="project:createBox($x, $y, $z)"/>
</xsl:template>
4. X3dToVml.html

- Handles DEF and USE
- Contains functions to button click events
5. Web Browser

IE v5.5 – displays incorrectly
IE v6.0 – displays correctly
Implementation Highlights

An UML diagram of all classes.

- Shape
  - Box
  - Cylinder
  - Cone
  - Sphere
  - Vector
  - Point
  - Group
  - Intensity
  - Transform
  - ProjectedPolygon
  - TransformationMatrix
1. Generate points in 3D for each primitive shape.
2. Transform 3D points.
3. Calculate color intensities.
4. Project points from 3D to 2D coordinate.
The primitive shapes are represented by points.

A box is represented by eight points.
These eight points are organized into a 2X4 array.

Box array:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P0</td>
<td></td>
<td>P1</td>
<td>P2</td>
<td>P3</td>
</tr>
<tr>
<td>row[1]</td>
<td>P4</td>
<td>P5</td>
<td>P6</td>
<td>P7</td>
</tr>
</tbody>
</table>
Wire frame form of the primitive shapes.
Implementation Highlights (cont.)

- Transformation
- Gouraud Shading
- The Phong Lighting model
- Projection
• The primitive shapes after shading.
Implementation Highlights (cont.)

User Interface.
Implementation Highlights (cont.)

History Buttons

History

0 1 2 3 4
M. L. M. U M. R. M. B. M. F.

Current position
Combine Transform tags if exceed 5
Implementation Highlights (cont.)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="X3d2Vml.xsl"?>
<!DOCTYPE X3D SYSTEM "X3D2VML.dtd">
<X3D>
  <Scene>
    <Transform rotation="0 0 1 0" scale="1 1 1" translation="-20 0 0">
      <Transform rotation="0 0 1 0" scale="1 1 1" translation="-20 0 0">
        <Transform rotation="0 0 1 0" scale="1 1 1" translation="-20 0 0">
          <Transform rotation="0 0 1 0" scale="1 1 1" translation="-20 0 0">
            <Transform rotation="1 0 0 0" scale="1 1 1" translation="0 0 0">
              <Shape>
                <Cylinder height="90" radius="70"/>
              </Shape>
            </Transform>
          </Transform>
        </Transform>
      </Transform>
    </Transform>
  </Scene>
</X3D>
```
# Maximum Load

<table>
<thead>
<tr>
<th># of Polygons</th>
<th>Time for changing perspective (millisecond)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1st click</td>
</tr>
<tr>
<td>12</td>
<td>47</td>
</tr>
<tr>
<td>72</td>
<td>312</td>
</tr>
<tr>
<td>144</td>
<td>2047</td>
</tr>
<tr>
<td>156</td>
<td>3562</td>
</tr>
</tbody>
</table>

A box has 12 polygons.
A cylinder has 72 polygons.
Limitations

• could not use the calculations done in the previous step.

• could not generate VML tags directly from the JavaScript section.
Conclusion

- Developed a X3D to VML translator
- Future work: clipping at object and polygon level
References

•[HD00] Beginning XML. David Hunter, with Curt Cagle, Dave Gibbons, Nikola Ozu, Jon Pinnock, Paul Spencer. Wrox Press Ltd. 2000
•[W3C97] Extensible Markup Language (XML). http://www.w3.org/XML. W3C.