3D Web Graphics without Plugins using VML

Student: Jiewei Lin

Advisor: Dr. Chris Pollett

Committee Members: Dr. Sin-Min Lee

Dr. Ho Kuen Ng

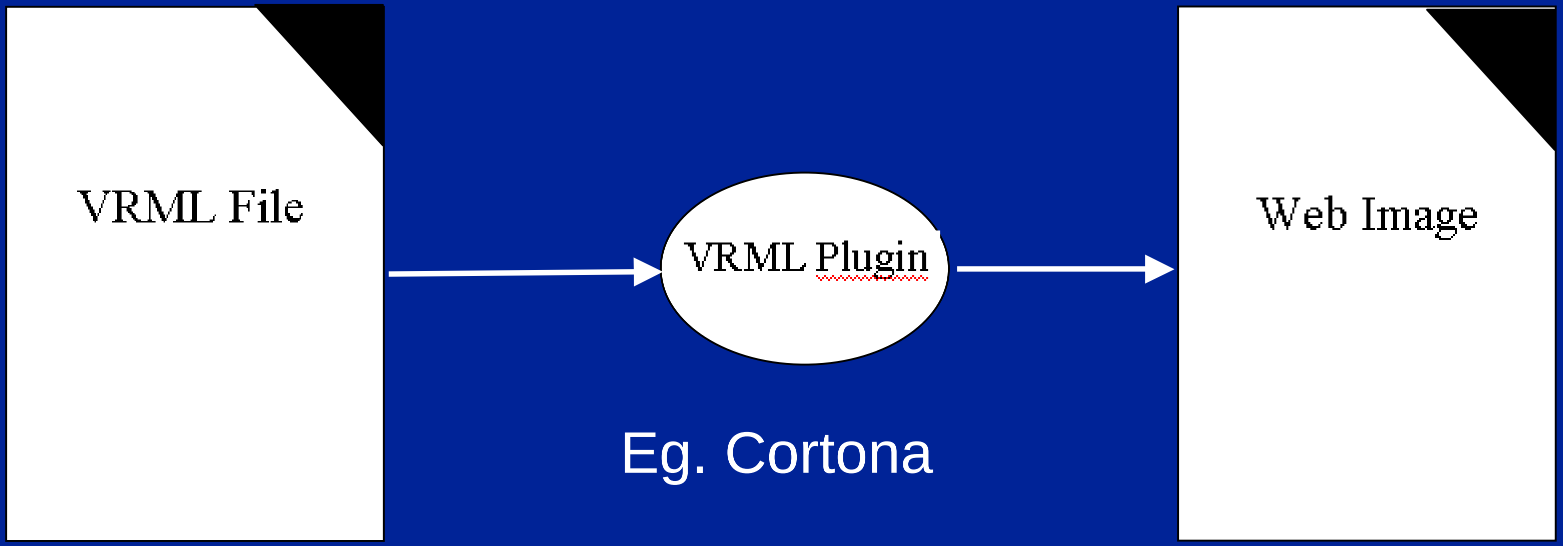
May 20, 2003

Overview

- Introduction
- Deployment requirements
- Implementation highlights
- Maximum Load
- Limitations
- Conclusion

Introduction

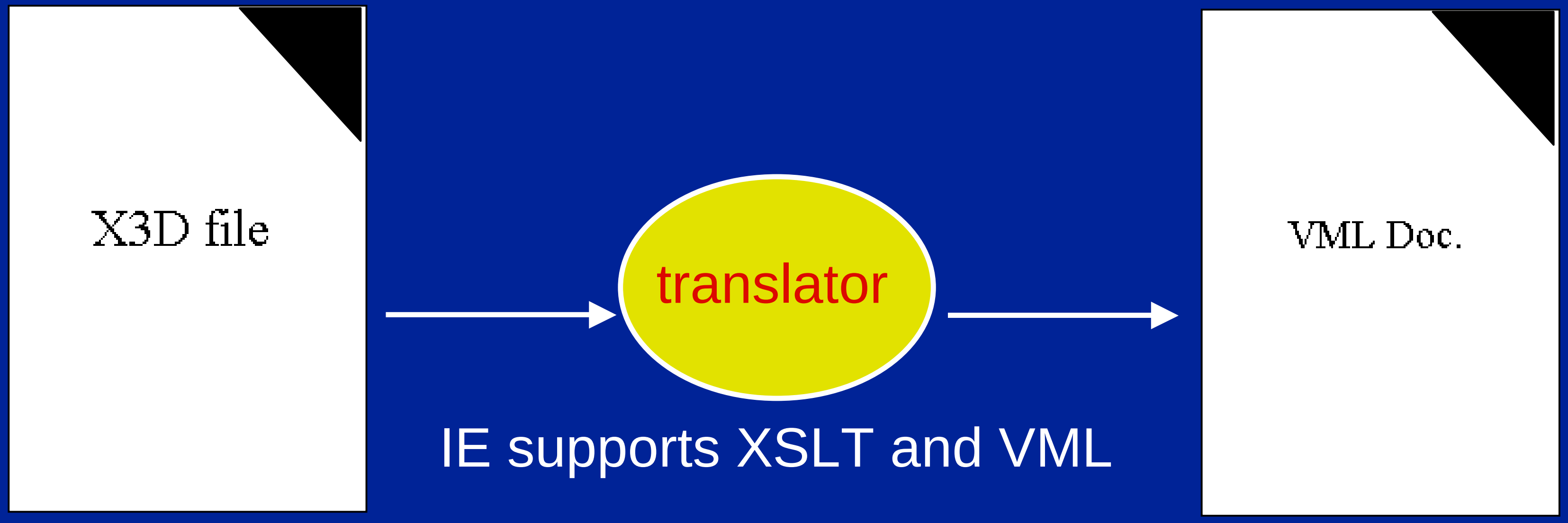
A VRML plugin is required to view VRML documents in Internet Explorer.



Eg. Cortona

Introduction (cont.)

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

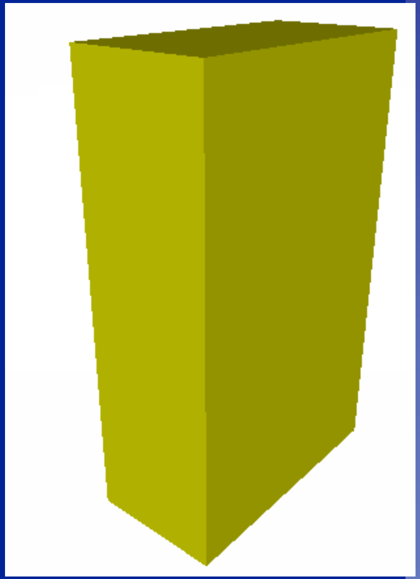


IE supports XSLT and VML

Introduction (cont.)

An X3D document.

```
<?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>
```

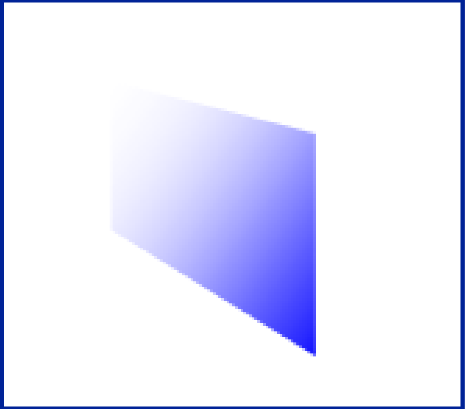


Introduction (cont.)

An VML document.

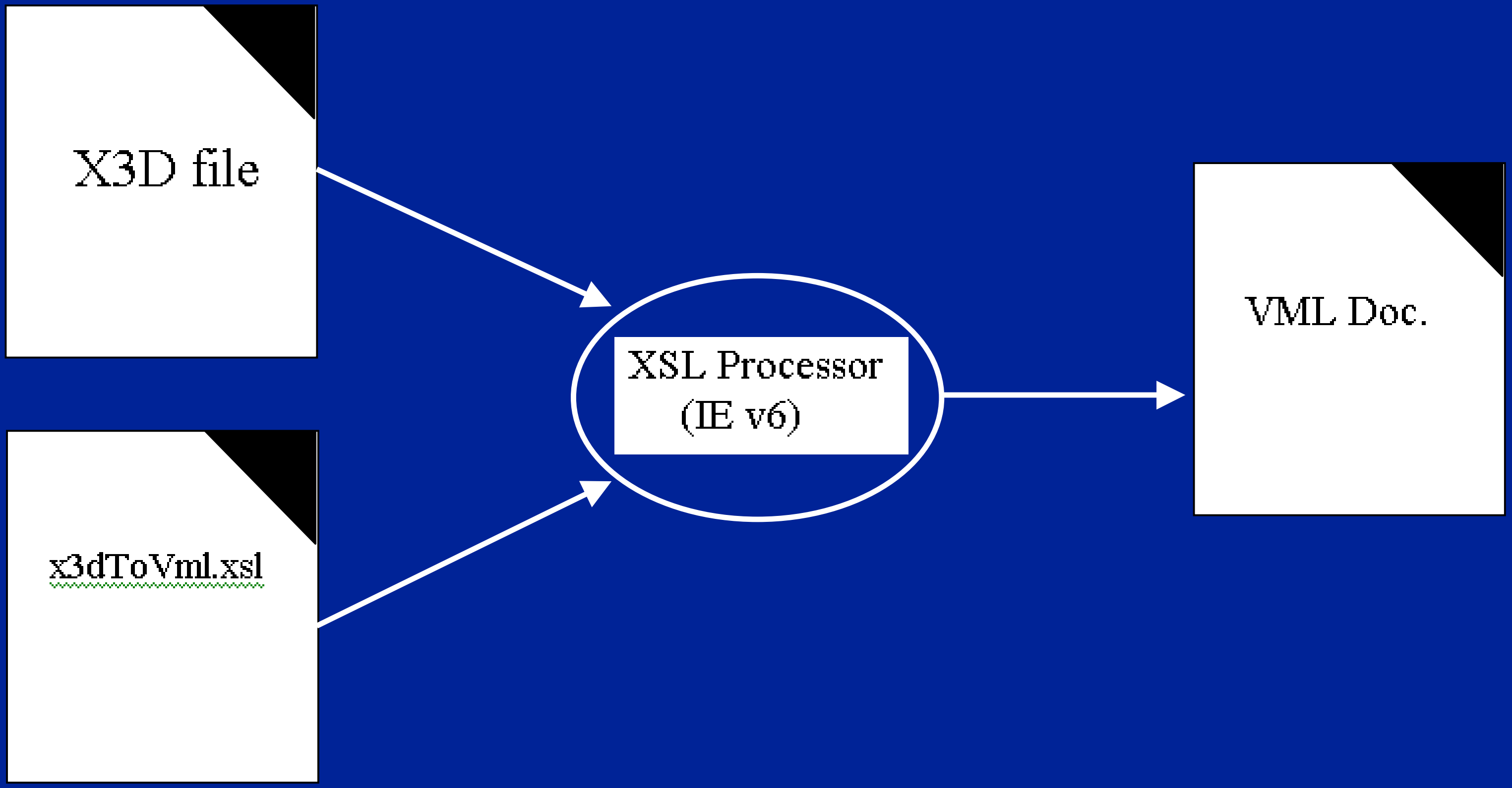
```
<html xmlns:v="urn:schemas-microsoft-com:vml"
xmlns:o="urn:schemas-microsoft-com:office:office"
xmlns="http://www.w3.org/TR/REC-html40">

<head>
<style>
v\:* {behavior:url(#default#VML);}
</style>
</head>
<body>
<v:polyline print="false“ points="181pt,154pt,126pt,140pt,126pt,
   180pt,181pt,214pt,181pt,154pt“ fill="true“ fillcolor="blue">
<v:stroke on="false"/>
<v:fill method="linear sigma" angle="45" type="gradient" />
</v:polyline>
</body>
</html>
```



Introduction (cont.)

An XSL Stylesheet



Requirements

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

Requirements (cont.)

1. A sample X3D input document

```
<?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>
```

Requirements (cont.)

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”

Requirements (cont.)

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">
```

Requirements (cont.)

3a. X3dToVml.xsl: JavaScript section

```
<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>
```

Requirements (cont.)

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>
```

Requirements (cont.)

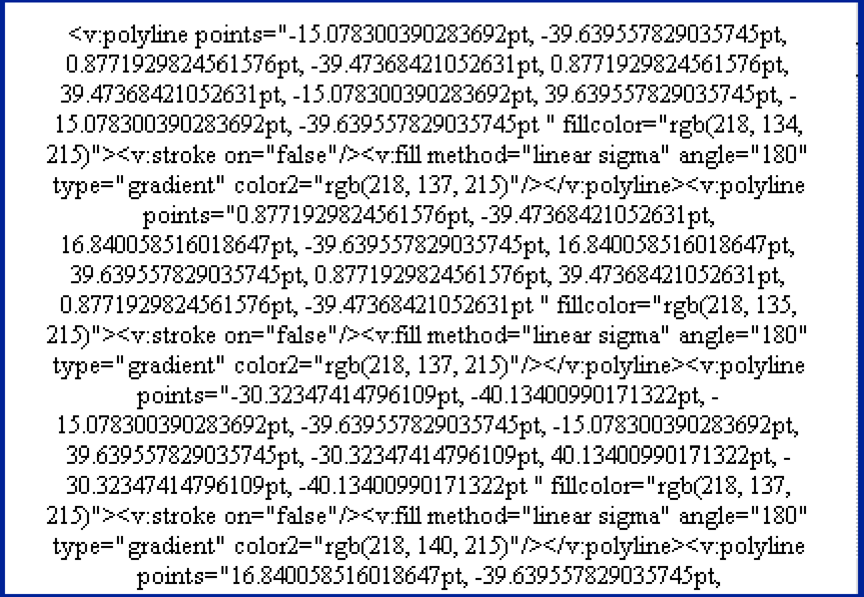
4. X3dToVml.html

- Handles DEF and USE
- Contains functions to button click events

Requirements (cont.)

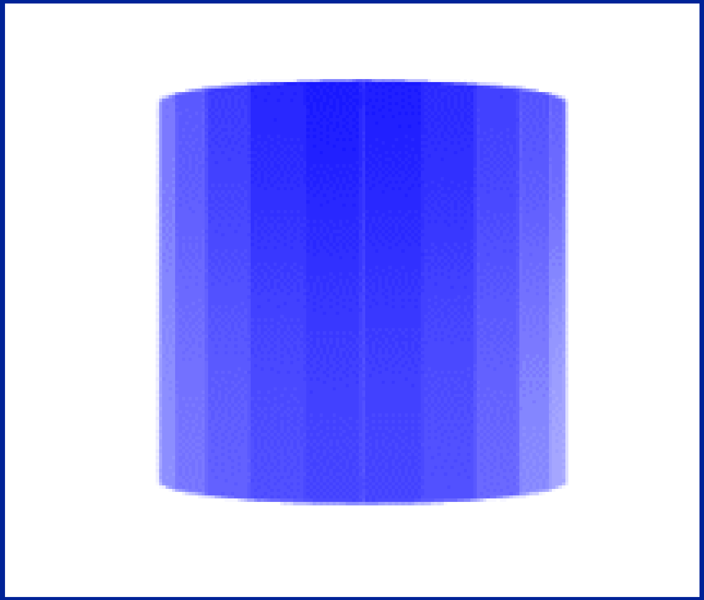
5. Web Browser

IE v5.5 – displays incorrectly



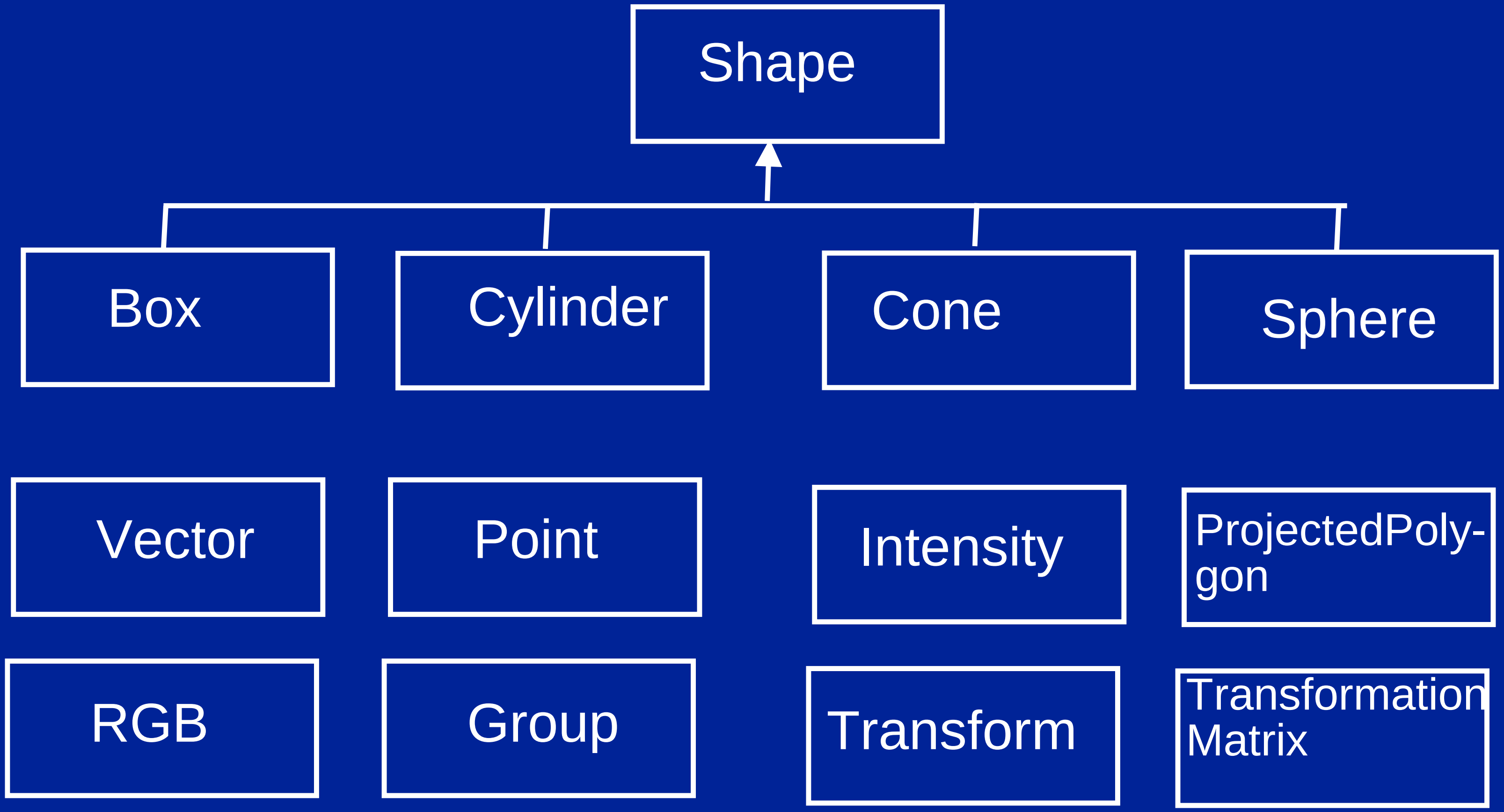
```
<v:polyline points="-15.078300390283692pt, -39.639557829035745pt,
0.8771929824561576pt, -39.47368421052631pt, 0.8771929824561576pt,
39.47368421052631pt, -15.078300390283692pt, 39.639557829035745pt, -
15.078300390283692pt, -39.639557829035745pt " fillcolor="rgb(218, 134,
215)"><v:stroke on="false"/><v:fill method="linear sigma" angle="180"
type="gradient" color2="rgb(218, 137, 215)"/></v:polyline><v:polyline
points="0.8771929824561576pt, -39.47368421052631pt,
16.840058516018647pt, -39.639557829035745pt, 16.840058516018647pt,
39.639557829035745pt, 0.8771929824561576pt, 39.47368421052631pt,
0.8771929824561576pt, -39.47368421052631pt " fillcolor="rgb(218, 135,
215)"><v:stroke on="false"/><v:fill method="linear sigma" angle="180"
type="gradient" color2="rgb(218, 137, 215)"/></v:polyline><v:polyline
points="-30.32347414796109pt, -40.13400990171322pt, -
15.078300390283692pt, -39.639557829035745pt, -15.078300390283692pt,
39.639557829035745pt, -30.32347414796109pt, 40.13400990171322pt, -
30.32347414796109pt, -40.13400990171322pt " fillcolor="rgb(218, 137,
215)"><v:stroke on="false"/><v:fill method="linear sigma" angle="180"
type="gradient" color2="rgb(218, 140, 215)"/></v:polyline><v:polyline
points="16.840058516018647pt, -39.639557829035745pt,
```

IE v6.0 – displays correctly



Implementation Highlights

An UML diagram of all classes.

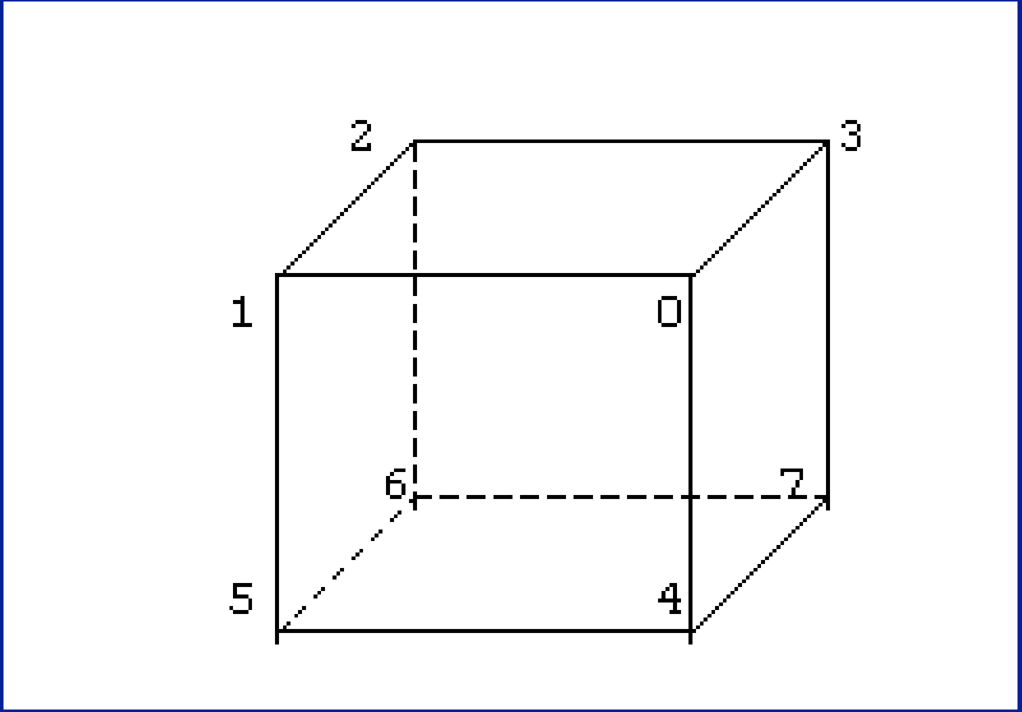


Implementation Highlights (cont.)

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.*

Implementation Highlights (cont.)

The primitive shapes are represented by points.



A box is represented by eight points.

Implementation Highlights (cont.)

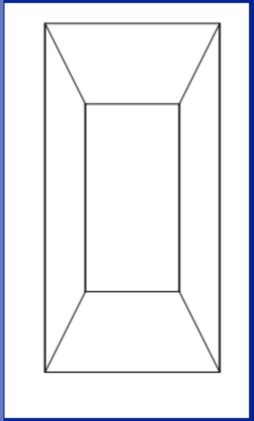
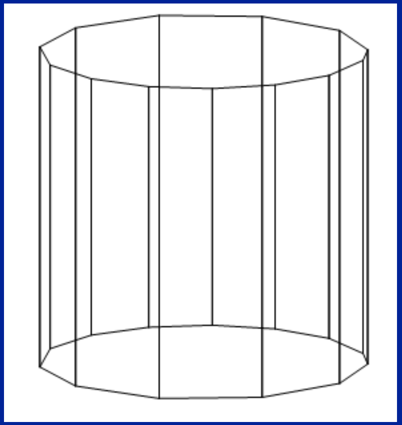
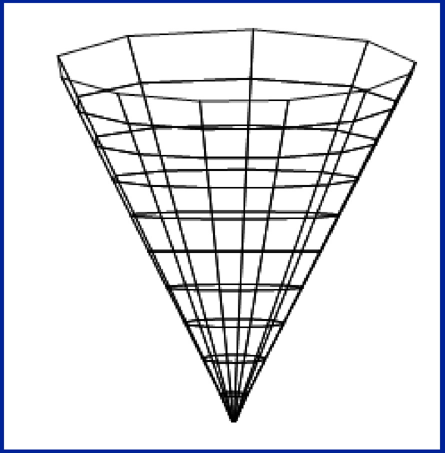
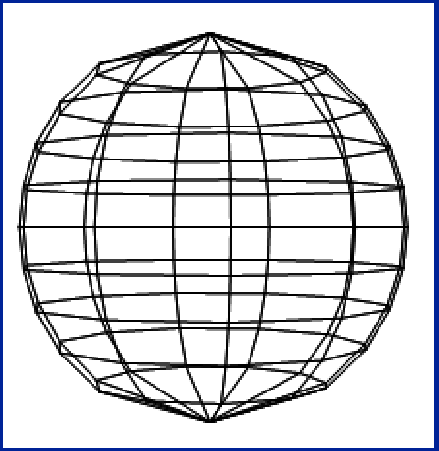
These eight points are organized into a 2X4 array.

Box array:

	Column[0]	Column[1]	Column[2]	Column[3]
row[0]	P0	P1	P2	P3
row[1]	P4	P5	P6	P7

Implementation Highlights (cont.)

Wire frame form of the primitive shapes.

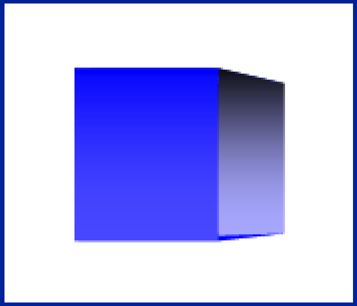
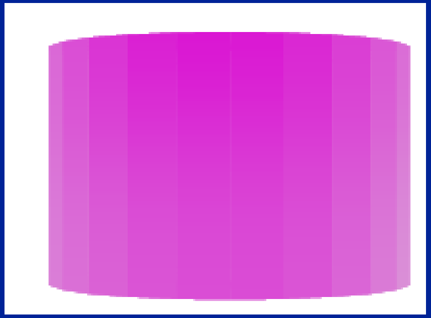
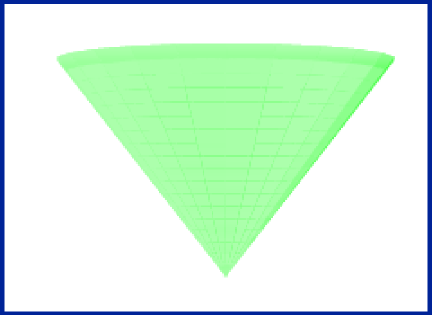
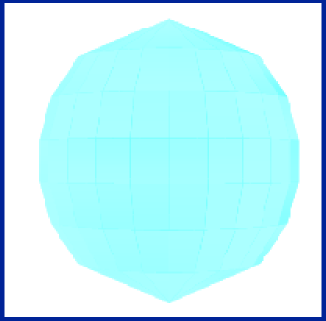





Implementation Highlights (cont.)

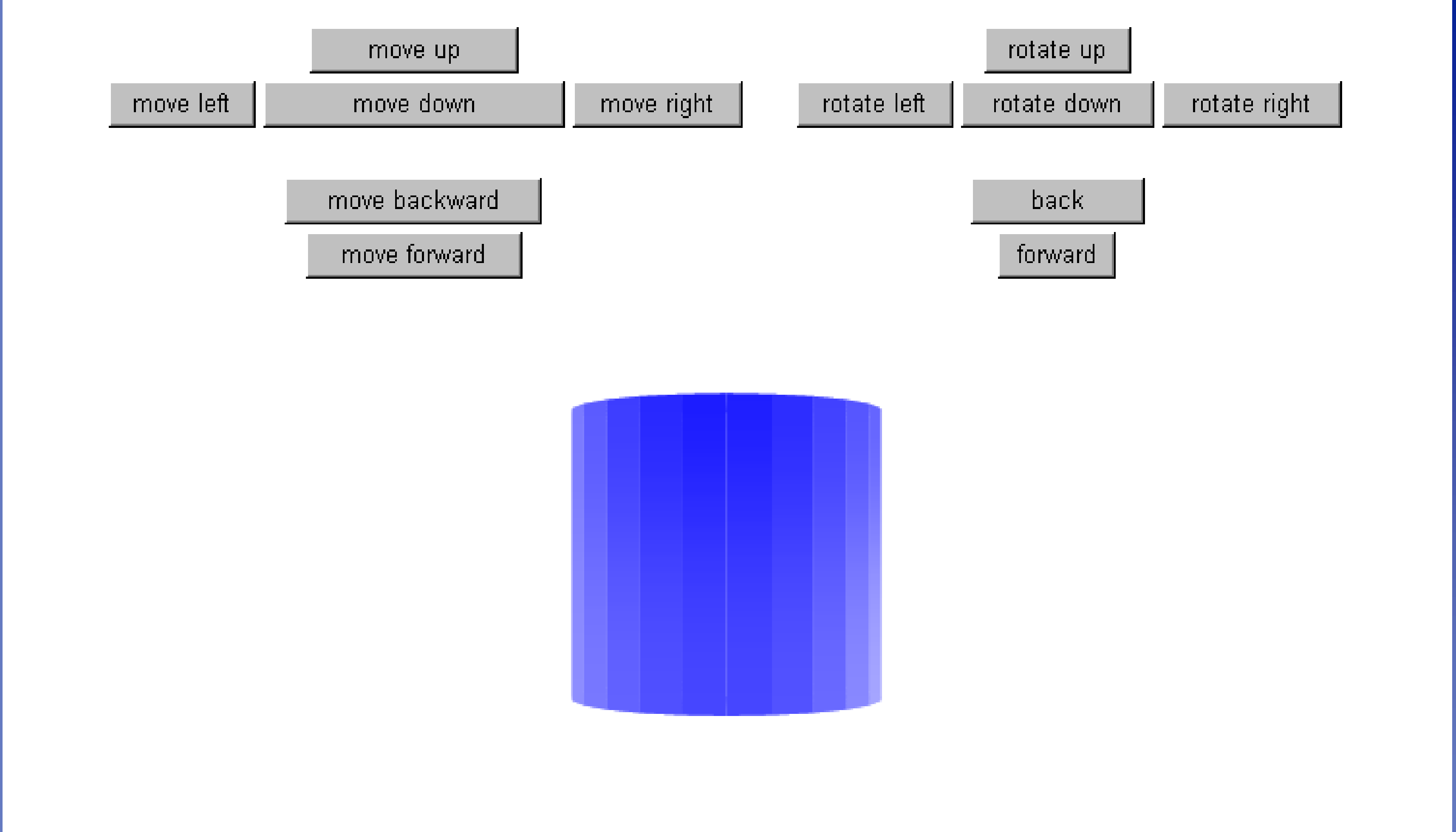
- Transformation
- Gouraud Shading
- The Phong Lighting model
- Projection

Implementation Highlights (cont.)

- The primitive shapes after shading.

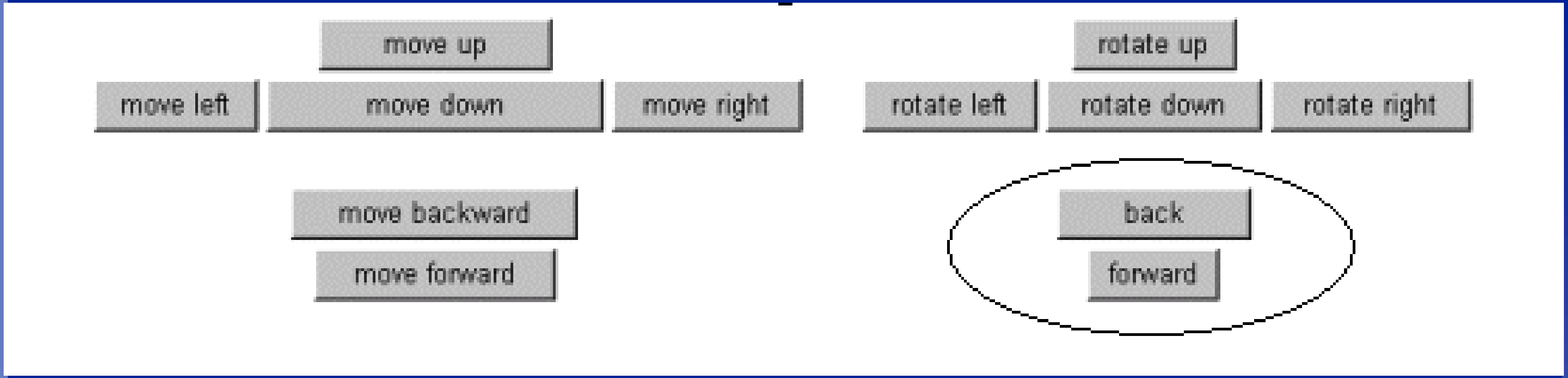





Implementation Highlights (cont.)

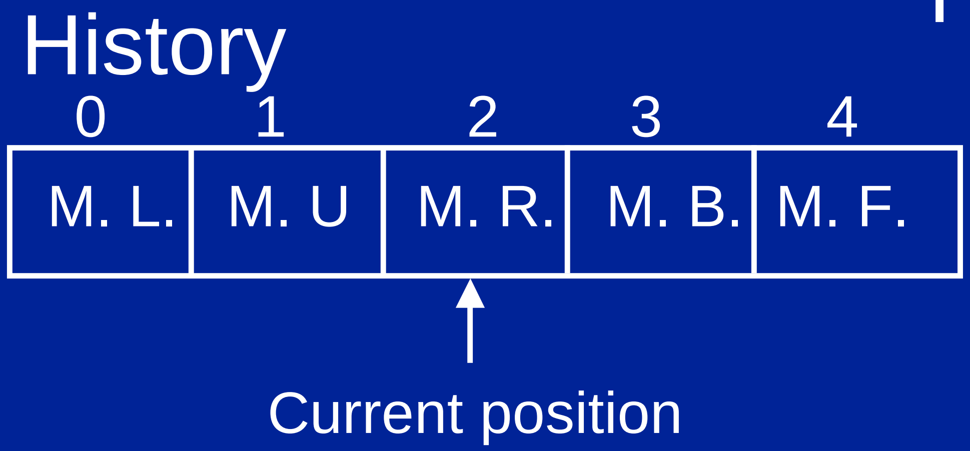


User Interface.

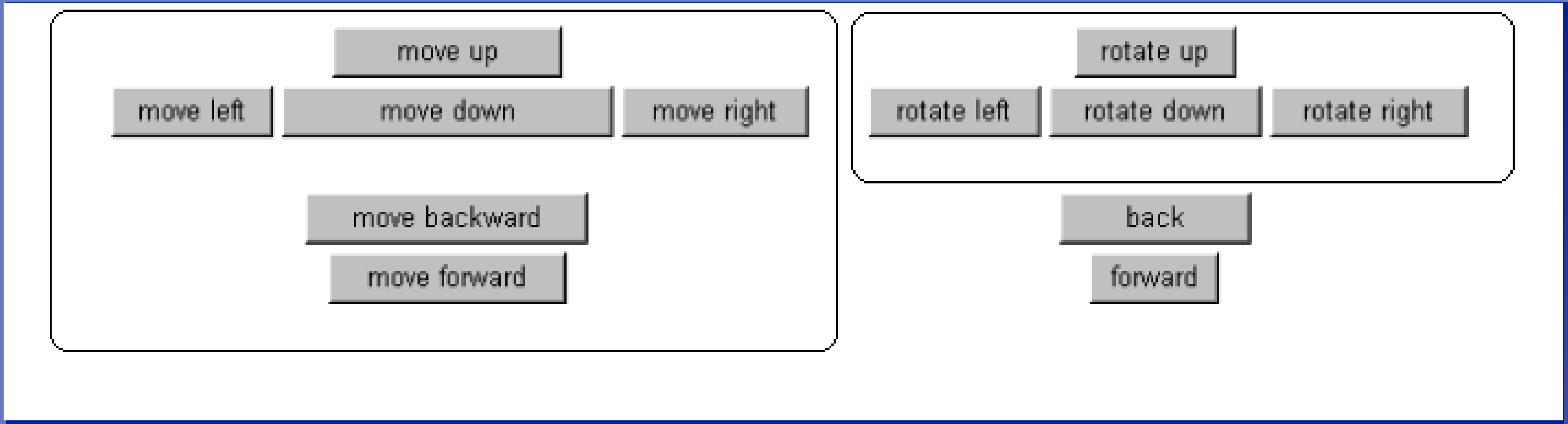
Implementation Highlights (cont.)



History Buttons



Implementation Highlights (cont.)



Combine Transform tags if exceed 5

Implementation Highlights (cont.)

```
<?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="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>
    </Transform>
  </Scene>
</X3D>
```

transformation tags

combined Transform tags

Maximum Load

# of Polygons	Time for changing perspective (millisecond)							
	1st click	2nd click	3rd click	4th click	5th click	6th click	7th click	8th click
12	47	47	47	47	47	47	47	47
72	312	328	343	343	344	344	344	344
144	2047	2078	2094	2109	2093	2125	2109	2110
156	3562	3657	3656	3766	3719	3719	3703	3719

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

•[ANM97] VRML 2.0 Sourcebook. 2nd edition. A. Ames, D. Nadeau, J. Moreland. Wiley. 1997.

•[ECMA99] Standard ECMA-262 ECMAScript Language Specification 3rd ed.

•http://www.ecma.ch/ecma1/stand/ecma-262.htm. ECMA. 1999.

•[FD98] JavaScript: The Definitive Guide. 3rd edition. David Flanagan. O'Reilly & Associates. 1998.

•[GR00] 3D Interactive VML. http://www.gersolutions.com/vml/. Gareth Richards. 2000.

•[HB97] Computer Graphics C Version. 2nd edition. Donald Hearn, M. Pauline Baker. Prentic Hall. 1997.

•[HD00] Beginning XML. David Hunter, with Curt Cagle, Dave Gibbons, Nikola Ozu, Jon Pinnock, Paul Spencer. Wrox Press Ltd. 2000

•[HR01] Xml Bible. 2nd edition. Elliotte Rusty Harold. Hungry Minds, Inc. 2001.

•[KM01] XSLT Programmer's Reference. 2nd edition. Michael Kay. Wrox Press Ltd. 2001.

•[MK01] LiveGraphics3D Documentation.

•http://wwwvis.informatik.uni-stuttgart.de/~kraus/LiveGraphics3D/documentation.html. Martin Kraus. 2001.

•[PS] 3D Web Graphics without Plugins using SVG. Paungkaew Sangtrakulcharoen. http://www.cs.sjsu.edu/faculty/pollett/masters/.

•[RE01] Learning XML. Erik T. Ray. 1st edition. O'Reilly & Associates. 2001.

•[SRB] 3D Computer Graphics A Mathematical Introduction with OpenGL. Samuel R. Buss. 2002.

•[W00] 3D Computer Graphics. 3rd ed. Allan Watt. Pearson Education Limited. (Addison Wesley). 2000.

•[W3C01] Scalable Vector Graphics (SVG) Specification 1.0. http://www.w3.org/TR/SVG/. W3C.

•[W3C97] Extensible Markup Language (XML). http://www.w3.org/XML. W3C.

•[W3C98b] VML - the Vector Markup Language. http://www.w3.org/TR/NOTE-VML. W3C.

•[W3C99] XSL Transformations (XSLT) Version 1.0. http://www.w3.org/TR/xslt. W3C.

•[W3DC01] Extensible 3D (X3D) Graphics Working Group. http://www.web3d.org/x3d.html. Web 3D Consortium. 2001.

Questions

?