

X3D Graphics for Web Authors

Chapter 6

Points, Lines and Polygons

Drawing is a struggle between nature and the artist, in which the better the artist understands the intentions of nature, the more easily he will triumph over it. For him it is not a question of copying, but of interpreting in a simpler and more luminous language.

Charles Baudelaire, *On the Ideal and the Model*, 1846.

Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary

References

Chapter Overview

Many different geometry nodes

An excellent aspect of X3D is that there are many different ways to create geometry

- Chapter 2, Geometry Primitives
- Chapter 6, Points, Lines and Polygons
- Chapter 10, Geometry2D Nodes
- Chapter 13, Triangles and Quadrilaterals

Fundamental geometry nodes

Geometry nodes in this chapter include points, lines, and indexed face sets

These nodes fundamental and can represent almost any shape

- Tools can convert other geometry to simpler forms
- Thus most are part of Interchange profile for broadest possible usage and adaptability

Overview: Points, Lines and Polygons

Triangles, single-sided polygons, normal vectors

Common fields: *ccw*, *convex*, *creaseAngle*, etc.

Geometry nodes, part 2

- Color and ColorRGBA
- Coordinate and CoordinateDouble
- PointSet
- IndexedLineSet and LineSet
- IndexedFaceSet
- ElevationGrid
- Extrusion

[back to Table of Contents](#)

Concepts

Triangles

Triangles are the primary low-level geometry construct used by graphics software, hardware

- More complex shapes are reduced to triangles by the rendering software
- A triangle is always planar, allowing the material appearance to fill it

Sometimes quadrilaterals are used, but problem is that values might be non-coplanar due to roundoff (or authoring) error

- Which means that filling in material is ill defined, and not properly or repeatably renderable

Single-sided polygons

Graphics engines always prefer simplicity in order to achieve maximum run-time performance

- Top 3 graphics hardware considerations: performance, performance, performance

Single-sided polygons take about half the time to draw than double-sided polygons

- So if authors can arrange geometry so that only one side is ever visible, can go single-sided
- Technical term: *backface culling*
- Efficiency is rationale for many X3D default values
- Example: default setting is *solid*='true' for X3D geometry primitives

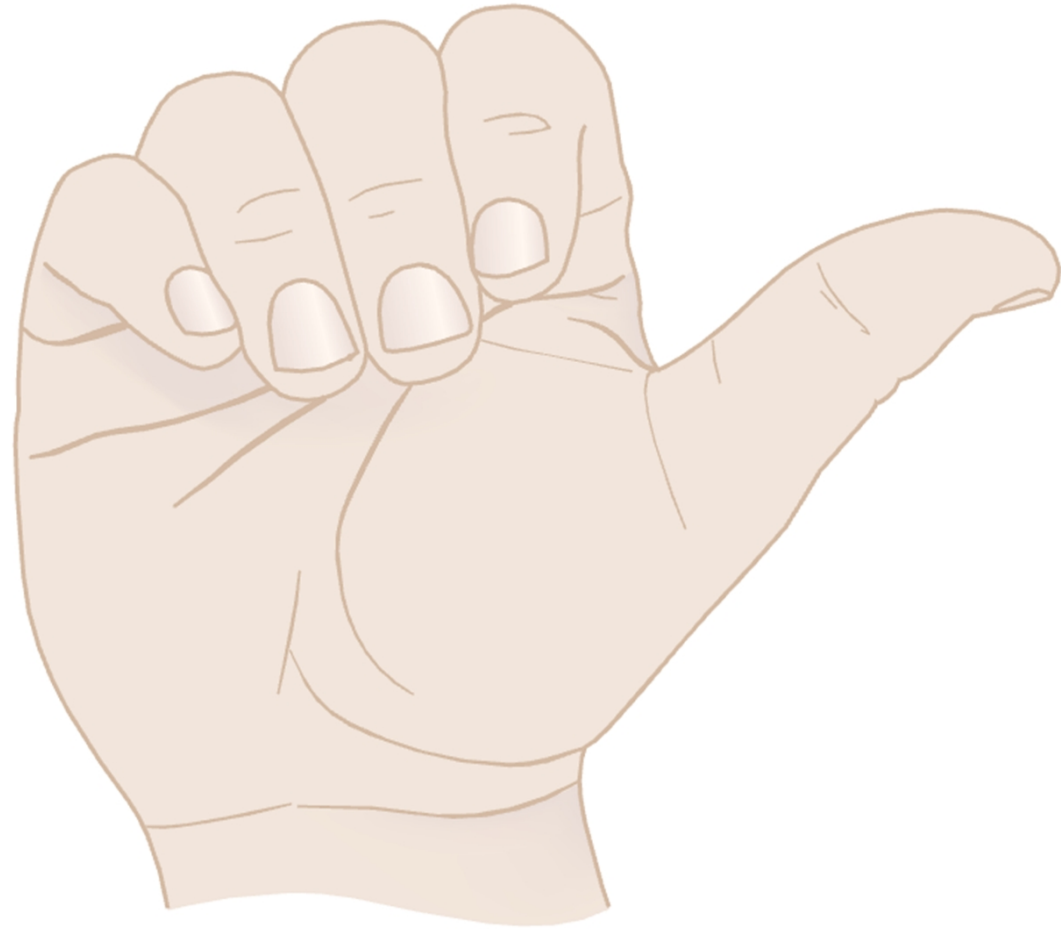
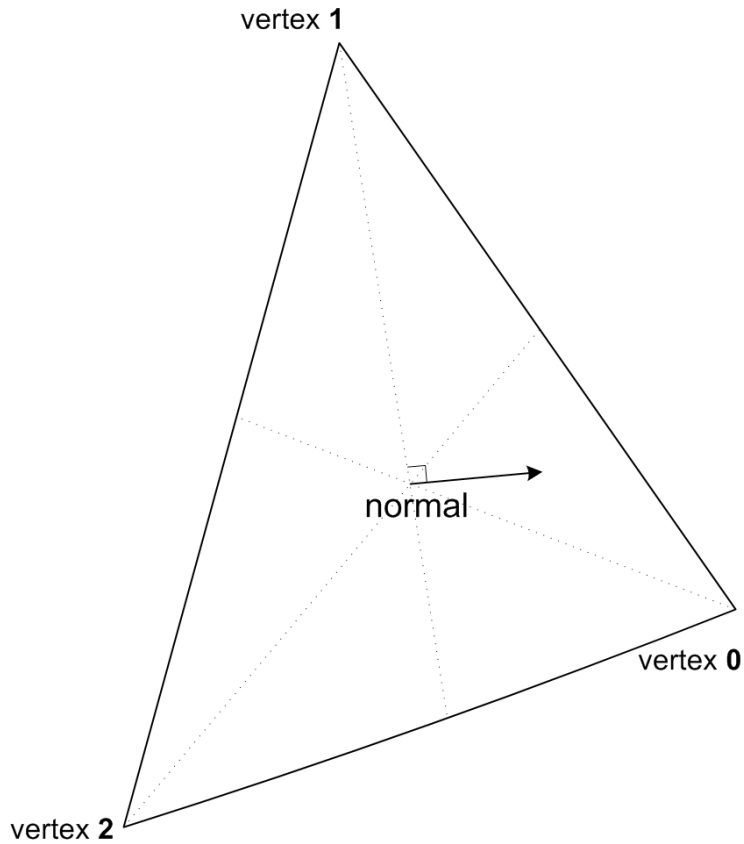
Normal vectors

The *normal* vector is perpendicular to the face, pointing away from the centroid of polygon

Direction of normal vector defined by order of points defining the polygon and right-hand rule

- Align curvature of fingers to match polygon vertex points 0, 1, 2 ...
- Thumb points in direction of (positive) normal, which is the front-facing side
- Negative normal thus points in direction of backface

Right-hand rule for polygon normals



normal vector is at polygon centroid, with perpendicular direction according to right-hand rule

Common field: *ccw*

ccw (counter clockwise) indicates whether default normal direction of polygons is counterclockwise (default) or clockwise

- *ccw*='true' is right-hand rule
- *ccw*='false' is opposite

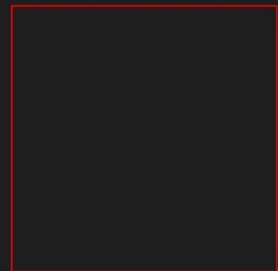
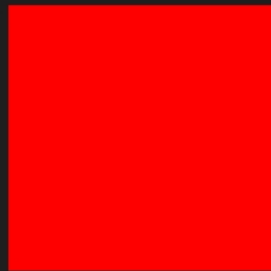
Hint: can correct some opposite-rendering geometry by reversing *ccw* value, rather than reordering all coordinates or indices

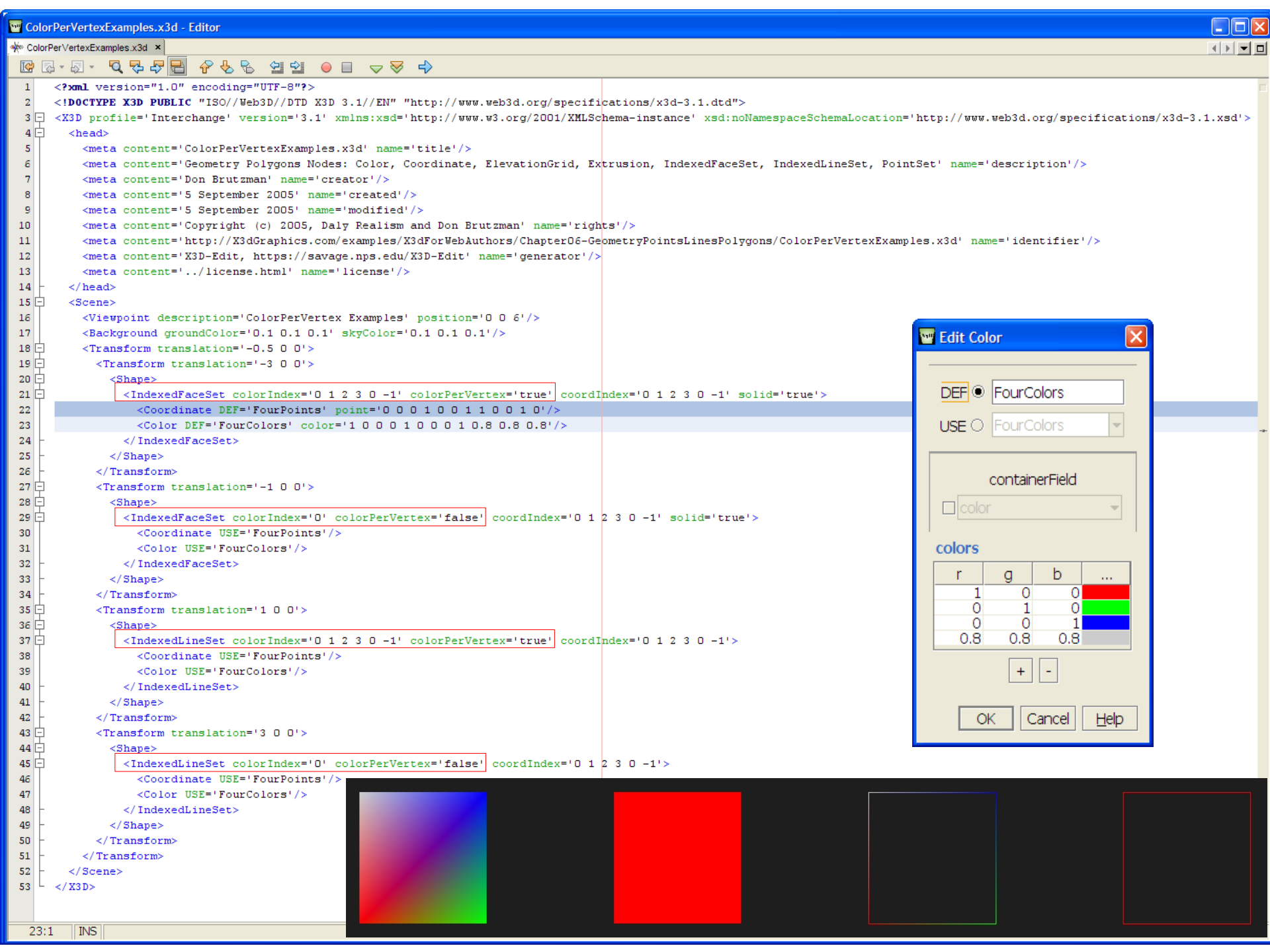
- Big time saver for some imports

Common field: *colorPerVertex*

colorPerVertex indicates whether contained color values are applied to each vertex point (default), or to each polygonal face

- *colorPerVertex*='true' requires that # colors must equal # points
- *colorPerVertex*='false' requires that # colors must equal # polygons





```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Interchange' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d-3.1.xsd'>
4   <head>
5     <meta content='ColorPerVertexExamples.x3d' name='title' />
6     <meta content='Geometry Polygons Nodes: Color, Coordinate, ElevationGrid, Extrusion, IndexedFaceSet, IndexedLineSet, PointSet' name='description' />
7     <meta content='Don Brutzman' name='creator' />
8     <meta content='5 September 2005' name='created' />
9     <meta content='5 September 2005' name='modified' />
10    <meta content='Copyright (c) 2005, Daly Realism and Don Brutzman' name='rights' />
11    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ColorPerVertexExamples.x3d' name='identifier' />
12    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
13    <meta content='../license.html' name='license' />
14  </head>
15  <Scene>
16    <Viewpoint description='ColorPerVertex Examples' position='0 0 6' />
17    <Background groundColor='0.1 0.1 0.1' skyColor='0.1 0.1 0.1' />
18    <Transform translation='-0.5 0 0'>
19      <Transform translation='-3 0 0'>
20        <Shape>
21          <IndexedFaceSet colorIndex='0 1 2 3 0 -1' colorPerVertex='true' coordIndex='0 1 2 3 0 -1' solid='true'>
22            <Coordinate DEF='FourPoints' point='0 0 0 1 0 0 1 1 0 0 1 0' />
23            <Color DEF='FourColors' color='1 0 0 0 1 0 0 0 1 0.8 0.8 0.8' />
24          </IndexedFaceSet>
25        </Shape>
26      </Transform>
27    <Transform translation='-1 0 0'>
28      <Shape>
29        <IndexedFaceSet colorIndex='0' colorPerVertex='false' coordIndex='0 1 2 3 0 -1' solid='true'>
30          <Coordinate USE='FourPoints' />
31          <Color USE='FourColors' />
32        </IndexedFaceSet>
33      </Shape>
34    </Transform>
35    <Transform translation='1 0 0'>
36      <Shape>
37        <IndexedLineSet colorIndex='0 1 2 3 0 -1' colorPerVertex='true' coordIndex='0 1 2 3 0 -1'>
38          <Coordinate USE='FourPoints' />
39          <Color USE='FourColors' />
40        </IndexedLineSet>
41      </Shape>
42    </Transform>
43    <Transform translation='3 0 0'>
44      <Shape>
45        <IndexedLineSet colorIndex='0' colorPerVertex='false' coordIndex='0 1 2 3 0 -1'>
46          <Coordinate USE='FourPoints' />
47          <Color USE='FourColors' />
48        </IndexedLineSet>
49      </Shape>
50    </Transform>
51  </Scene>
52 </X3D>
```

Edit Color

DEF ☒ FourColors

USE ☐ FourColors

containerField

☐ color

colors

r	g	b	...
1	0	0	
0	1	0	
0	0	1	
0.8	0.8	0.8	

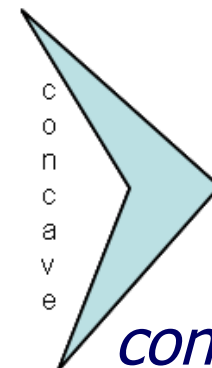
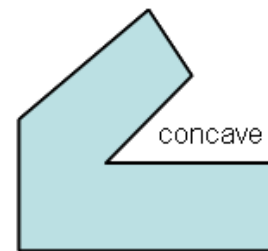
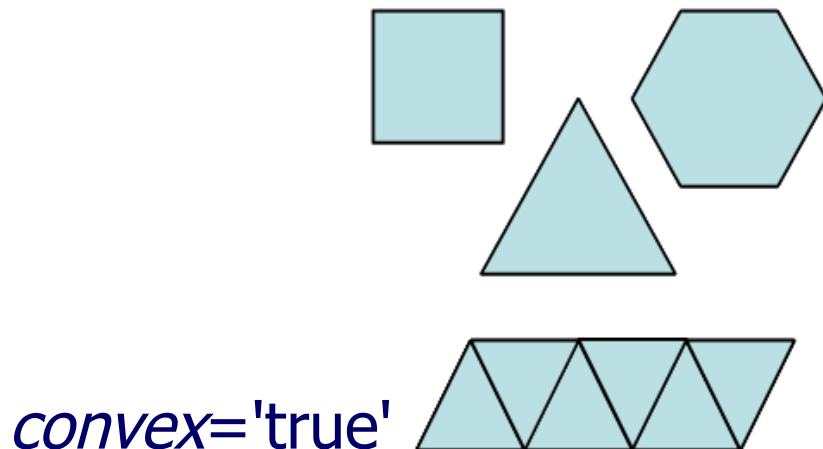
+ -

OK Cancel Help

Common field: *convex*

convex indicates whether an n -sided polygon has concave sides, meaning empty-space cavities

- *convex*='true' (default) means no concave sides
- *convex*='false' means concave sides may exist in the polygon, so extra care is needed to avoid hardware or software difficulty when rendering



convex='false'

Common field: *creaseAngle*

creaseAngle defines the angle (in radians) used to determine whether adjacent polygons are drawn with sharp edges or smooth shading

- If angle between polygons is less than *creaseAngle*, then smooth shading is used
- Smooth shading can conceal underlying tessellation

creaseAngle only affects shading within a single geometric shape, not exterior boundaries

- *creaseAngle*='0.0' means all edges are sharp
- *creaseAngle*='3.14159' means no edges are sharp

Common field: *normalPerVertex*

normalPerVertex indicates whether contained normal values are applied to each vertex point (default), or to each polygonal face

- *normalPerVertex*='true' requires that # normals must equal # points
- *normalPerVertex*='false' requires that # normals must equal # polygons

Common index fields:

colorIndex, coordIndex, normalIndex

colorIndex, coordIndex, normalIndex

each provide arrays of integer indices that correlate Color, Coordinate or Normal values to individual vertices or polygons

- Initial index is 0
- Sentinel value -1 concludes polygon, polyline
- Maximum value equals (count - 1)
- Type MFInt32, default is empty array

index counting checks

- *colorIndex* must equal (point count - 1) when *colorPerVertex*='true', which is default
- *colorIndex* must equal (polygon count - 1) when *colorPerVertex*='false'
- *normalIndex* must equal (point count - 1) when *normalPerVertex*='true', which is default
- *normalIndex* must equal (polygon count - 1) when *normalPerVertex*='false'

X3D Nodes and Examples

Color node

Color values for individual polygons, line segments and points can be defined using the Color node

Color values are red-green-blue (RGB) [0..1]

- Type is MFColor array of 3-tuple values
- HTML, SVG colors are [0..255] [#000000..#FFFFFF] and so must be converted numerically if used

Appearance and Material node can also be used to control overall transparency, if needed

ColorRGBA node

ColorRGBA used similarly to Color node, but adds alpha (opacity) to each red-green-blue value

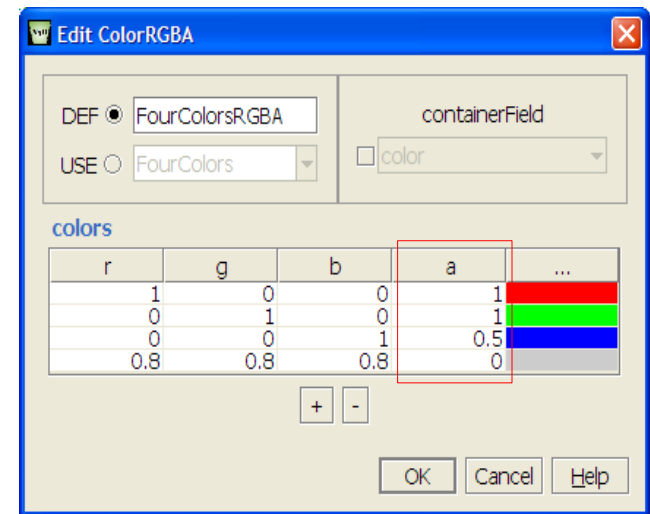
- alpha component equals (1 – transparency)

Alpha values range 0 to 1

- 0 means fully transparent
- 1 means fully opaque

RGBA values selectively allow transparent parts in geometry

- Rather than single Material transparency consistently across full geometry with Color node



Color.x3d x

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/s
4 <head>
5   <meta content='Color.x3d' name='title' />
6   <meta content='Vertex color applied to IndexedFaceSet built positive-displacement cylinder pump house. Note that this scene is not really
7     static because the PositionInterpolators contain an initial offset of the piston and rocker arm.' name='description' />
8   <meta content='Todd Gagnon and Mark A. Boyd' name='authors' />
9   <meta content='Xeena VRML importer' name='translator' />
10  <meta content='8 June 1998' name='created' />
11  <meta content='20 December 2002' name='imported' />
12  <meta content='3 February 2007' name='modified' />
13  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/Color.x3d' name='identifier' />
14  <meta content='KelpTank.x3d' name='reference' />
15  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference' />
16  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
17  <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator' />
18  <meta content='../license.html' name='license' />
19 </head>
20 <Scene>
21   <WorldInfo info='DTG of last update: 081200Jun98 Added: Updated: cycleInterval to 5.5 Modeled by: Todd Gagnon and Mark A. Boyd' title='pumpHou
22   <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Cha
23   <ProtoInstance name='WhereAmI' />
24   <Background skyColor='1 1 1' />
25   <Viewpoint description='Book View' orientation='0 -1 0 0.53' position='-2.28 0.29 4.06' />
26   <Group>
27     <Shape>
28       <Appearance DEF='pumpHouse'>
29         <Material diffuseColor='0.82 0.78 0.74' />
30       </Appearance>
31       <IndexedFaceSet
32         ccw='true'
33         colorIndex='0 0 0 0 -1 0 0 1 1 -1 1 1 1 1 -1 1 0 0 1 -1 0 0 1 1 -1 1 1 1 -1 0 0 1 1 -1 0 0 1 1'
34         colorPerVertex='true'
35         coordIndex='0 1 5 4 -1 5 1 2 6 -1 6 2 3 7 -1 3 0 4 7 -1 1 12 13 2 -1 2 13 14 -1 12 15 16 13 -1
36         <Coordinate DEF='CoordinateNodeExample' point='0.0 0.0 0.0 2.0 0.0 0.0 2.0 0.0 0.0 2.0 1.75 0.0 0.0 1.75 0.0 0
37         <Color DEF='ColorNodeExample' color='.82 .78 .74 .66 .37 .02' />
38       </IndexedFaceSet>
39     </Shape>
40     <Transform scale='0.91 0.6 0.3' translation='0.8 -0.65 0.5'>
41       <Shape>
42         <Appearance>
43           <Material diffuseColor='0.749 0.694 0.651' />
44         </Appearance>
45         <Cylinder bottom='false' top='false' />
46       </Shape>

```

37:1 INS

Edit Color

DEF ☒ ColorNodeExample

USE ☐ ColorNodeExample

containerField

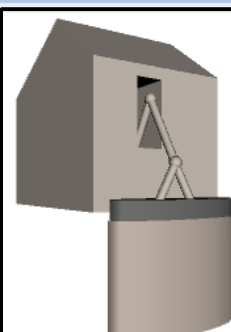
☐ color

colors


r	g	b	...
0.82	0.78	0.74	
0.66	0.37	0.02	

+ -

OK Cancel Help



Color	<p>Color node defines a set of RGB color values. Color is only used by ElevationGrid, IndexedFaceSet, IndexedLineSet, LineSet and PointSet.</p> <p>Hint: colors are often controlled by Material instead.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
color	<p>[color: accessType inputOutput, type MFCOLOR CDATA #IMPLIED]</p> <p>color defines a set of RGB colors.</p>
containerField	<p>[containerField: NMTOKEN "color"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

 ColorRGBA	<p>ColorRGBA node defines a set of RGBA color values. ColorRGBA is only used by ElevationGrid, IndexedFaceSet, IndexedLineSet, LineSet and PointSet.</p> <p>Hint: colors are often controlled by Material instead.</p> <p>Hint: alpha channel may be ignored under Interchange profile.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
color	<p>[color: accessType inputOutput, type MFCOLORRGBA CDATA #IMPLIED]</p> <p>color defines a set of RGBA colors.</p>
containerField	<p>[containerField: NMTOKEN "color"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

Coordinate node

Provide array of x-y-z point values

- Type MFVec3f array of 3-tuple values, each with 32-bit single-precision floating point

Coordinate *point* values define all of the vertices needed to build polygonal geometry

- *coordIndex* array indicates connectivity for each individual polygon in the shape
- *coordIndex* value -1 indicates end of one polygon, next *coordIndex* value indicates vertex point that begins a new polygon

CoordinateDouble node

Definition and usage similar to Coordinate node

Provide array of x-y-z point values

- Type MFVec3d array of 3-tuple values, each with 64-bit double-precision floating point

Double precision may be needed for specialty applications (geographic, atomic, etc.)

Note however that most graphics hardware is exclusively single-point precision, for speed

- So browser may need special software techniques to handle double precision fidelity properly

Color.x3d x

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/s
4 <head>
5   <meta content='Color.x3d' name='title' />
6   <meta content='Vertex color applied to IndexedFaceSet built positive-displacement cylinder pump house. Note that this scene is not really
7     static because the PositionInterpolators contain an initial offset of the piston and rocker arm.' name='description' />
8   <meta content='Todd Gagnon and Mark A. Boyd' name='authors' />
9   <meta content='Xeena VRML importer' name='translator' />
10  <meta content='8 June 1998' name='created' />
11  <meta content='20 December 2002' name='imported' />
12  <meta content='3 February 2007' name='modified' />
13  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/Color.x3d' name='identifier' />
14  <meta content='KelpTank.x3d' name='reference' />
15  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.'
16  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
17  <meta content='Vrml97ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator' />
18  <meta content='../license.html' name='license' />
19 </head>
20 <Scene>
21   <WorldInfo info='DTG of last update: 081200Jun98 Added: Updated: cycleInterval to 5.5 Model
22   <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' ht
23   <ProtoInstance name='WhereAmI' />
24   <Background skyColor='1 1 1' />
25   <Viewpoint description='Book View' orientation='0 -1 0 0.53' position='-2.28 0.29 4.06' />
26   <Group>
27     <Shape>
28       <Appearance DEF='pumpHouse'>
29         <Material diffuseColor='0.82 0.78 0.74' />
30       </Appearance>
31       <IndexedFaceSet
32         ccw='true'
33         colorIndex='0 0 0 0 -1 0 0 1 1 -1 1 1 1 1 -1 1 0 0 1 -1 0 0 1 1 -1 1 1 1 -1 0 0 1 1
34         colorPerVertex='true'
35         coordIndex='0 1 5 4 -1 5 1 2 6 -1 6 2 3 7 -1 3 0 4 7 -1 1 12 13 2 -1 2 13 14 -1 12
36         <Coordinate DEF='CoordinateNodeExample' point='0.0 0.0 0.0 2.0 0.0 0.0 2.0 1.75 0.0 0.0 1.75 0.0 0.625 0.75 0.0 1.0 0.75 0.0 1.0 1.6 0.0
37         <Color DEF='ColorNodeExample' color='.82 .78 .74 .66 .37 .02' />
38       </IndexedFaceSet>
39     </Shape>
40     <Transform scale='0.91 0.6 0.3' translation='0.8 -0.65 0.5'>
41       <Shape>
42         <Appearance>
43           <Material diffuseColor='0.749 0.694 0.651' />
44         </Appearance>
45         <Cylinder bottom='false' top='false' />
46       </Shape>

```

Edit Coordinate

DEF ☒ CoordinateNodeExample containerField

USE ☐ LateNodeExample ☐ coord

CoordinateDouble ☐

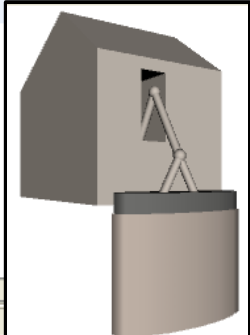
point

X	Y	Z
0	0	0
2	0	0
2	1.75	0
0	1.75	0
0.625	0.75	0

+ -

OK Cancel Help

37:1 INS



Coordinate	Coordinate builds geometry using a set of 3D coordinates. Coordinate is used by IndexedFaceSet, IndexedLineSet, LineSet and PointSet. Coordinate is also used by NurbsPositionInterpolator and NurbsOrientationInterpolator.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
point	[point: accessType inputOutput, type MFVec3f CDATA #IMPLIED] point contains a set of 3D coordinates.
containerField	[containerField: NMTOKEN "coord"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.
CoordinateDouble	CoordinateDouble builds geometry using a set of 3D coordinates. CoordinateDouble is used by IndexedFaceSet, IndexedLineSet, LineSet and PointSet. CoordinateDouble is also used by NurbsPositionInterpolator and NurbsOrientationInterpolator.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
point	[point: accessType inputOutput, type MFVec3d CDATA #IMPLIED] point contains a set of 3D coordinates.
containerField	[containerField: NMTOKEN "coord"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

PointSet node

PointSet creates a series of simple unconnected points in 3D space

- Contains Coordinate node for *point* data
- Since points are separate, *coordIndex* unnecessary

Each point typically drawn as a single pixel

- Or consistently as multiple pixels
- Thus scaling and perspective are quite deceiving
- **Rarely used** due to perspective inconsistencies

Color can be set in one of two ways

- Uniformly via Material *emissiveColor* value
- Individually via contained Color/ColorRGBA node

```

1 <?xml version='1.0' encoding='utf-8'?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/s
4 <head>
5   <meta content='PointSet.x3d' name='title' />
6   <meta content='Way points for the animated shark Lucy traversing the tank.' name='description' />
7   <meta content='Tim McLean' name='creator' />
8   <meta content='Don Brutzman' name='translator' />
9   <meta content='June 1998' name='created' />
10  <meta content='20 December 2002, 11 March 2006' name='modified' />
11  <meta content='http://web.nps.navy.mil/~brutzman/kelp' name='reference' />
12  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/SharkLucyLocale.x3d' name='reference' />
13  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/PointSet.x3d' name='identifier' />
14  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
15  <meta content='../license.html' name='license' />
16 </head>
17 <Scene>
18   <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Cha
19   <ProtoInstance name='WhereAmI' />
20   <Background skyColor='0 0 0' />
21   <Viewport description='Book View' orientation='0.939 0.335 0.075 -0.57' position='-0.89 1.91 9.26' />
22   <Transform DEF='_0' translation='0.0 -8.0 -1.0'>
23     <Inline url='SharkLucy.wrl' "http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/SharkLucy.wrl" "SharkLucy.x3d" "http://X3dG
24     <Group>
25       <TimeSensor DEF='SHARK1_CLOCK' cycleInterval='220.0' loop='true' />
26       <PositionInterpolator DEF='SHARK1_POSITION' key='0.0 0.048 0.112 0.155 0.1' keyValue='0.0 0.048 0.112 0.155 0.1' />
27       <OrientationInterpolator DEF='SHARK1_ORIENTATION' key='0.0 0.048 0.112 0.1' keyValue='0.0 0.048 0.112 0.1' />
28     </Group>
29   </Transform>
30   <Transform translation='0 0 0'>
31     <Shape>
32       <Appearance>
33         <Material emissiveColor='1 1 0' />
34       </Appearance>
35       <PointSet>
36         <Coordinate point='0.0 -7.0 -1.0 -1.75 -7.0 -0.5 -4.0 -7.0 0.5 -5.0 -6.5 1.5 -5.5 -6.25 0.75 -5.25 -5.5 -2.25 -4.25 -5.0 -3.25 -2.75 -4.
37       </PointSet>
38     </Shape>
39   </Transform>
40   <TimeSensor DEF='_4' loop='true' />
41   <Script DEF='sharkSwimmingInTankTrigger_5'>
42     <field accessType='inputOnly' name='triggerIn' type='SFTime' />
43     <field accessType='outputOnly' name='startTime' type='SFTime' />
44     <field accessType='outputOnly' name='firstTime' type='SFBool' />
45     <![CDATA[ecmascript:
46

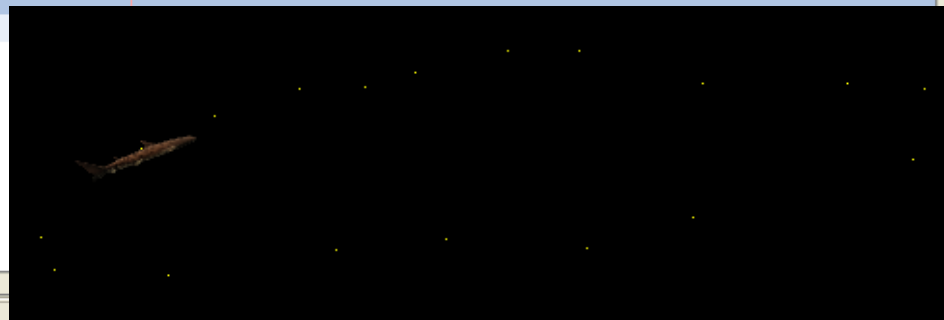
```


Edit PointSet

DEF ☒

USE ☐

OK Cancel Help



 PointSet	<p>PointSet is a node that contains a set of colored 3D points, represented by contained Color and Coordinate nodes. Color values or a Material emissiveColor is used to draw lines and points.</p> <p>Hint: use a different color (or emissiveColor) than the background color.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

IndexedLineSet node

IndexedLineSet creates an array of line segments

- Contains Coordinate node for *point* data
- Can be discontinuous or share points repeatedly
- Each set of connected line segments is a *polyline*

Lines are not lit, are not texture-mapped, and do not participate in collision detection

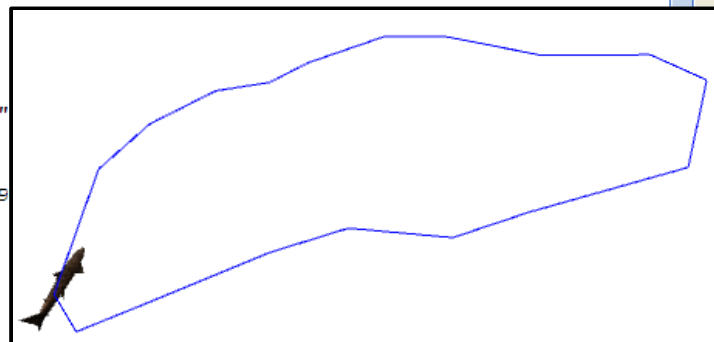
Color can be set in one of two ways

- Uniformly via Material *emissiveColor* value
- Individually via contained Color/ColorRGBA node; applied either by individual points, or by each segment, as determined by *colorPerVertex*

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
3 <X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/s
4 <head>
5   <meta content='IndexedLineSet.x3d' name='title' />
6   <meta content='The path of the animated shark Lucy traversing the tank.' name='description' />
7   <meta content='Tim McLean' name='creator' />
8   <meta content='Don Brutzman' name='translator' />
9   <meta content='June 1998' name='created' />
10  <meta content='3 February 2008' name='modified' />
11  <meta content='http://web.nps.navy.mil/~brutzman/kelp' name='reference' />
12  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/SharkLucyLocale.x3d' name='reference' />
13  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/IndexedLineSet.x3d' name='identifier' />
14  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
15  <meta content='../license.html' name='license' />
16 </head>
17 <Scene>
18   <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI'
19   <ProtoInstance name='WhereAmI' />
20   <Background skyColor='1 1 1' />
21   <Viewpoint description='Book View' orientation='0.939 0.335 0.075 -0.57' position='-0.89
22   <Transform translation='0 0 0'>
23     <Shape>
24       <Appearance>
25         <Material emissiveColor='0 0 1' />
26       </Appearance>
27       <IndexedLineSet DEF='ILS' coordIndex='0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 0 -1'>
28         <Coordinate point='0.0 -7.0 -1.0 -1.75 -7.0 -0.5 -4.0 -7.0 0.5 -5.0 -6.5 1.5 -5.5 -6.25 0.75 -5.25 -5.5 -2.25 -4.25 -5.0 -3.25 -2.75 -4.
29       </IndexedLineSet>
30     </Shape>
31   </Transform>
32   <Transform DEF='_0' translation='0.0 -8.0 -1.0'>
33     <Inline url='"SharkLucy.wrl"' http://X3dGraphics.com/examples/X3dForWebAuthors/KelpFore
34   </Group>
35     <TimeSensor DEF='SHARK1_CLOCK' cycleInterval='220.0' loop='true' />
36     <PositionInterpolator DEF='SHARK1_POSITION' key='0.0 0.048 0.112 0.155 0.184 0.263 0.3
37     <OrientationInterpolator DEF='SHARK1_ORIENTATIONS' key='0.0 0.048 0.112 0.155 0.184 0.2
38   </Group>
39 </Transform>
40 <TimeSensor DEF='_4' loop='true' />
41 <Script DEF='sharkSwimmingInTankTrigger_5'>
42   <field accessType='inputOnly' name='triggerIn' type='SFTime' />
43   <field accessType='outputOnly' name='startTime' type='SFTime' />
44   <field accessType='outputOnly' name='firstTime' type='SFBool' />
45   <![CDATA[ecmascript:
46

```



Edit IndexedLineSet

DEF ☒ ILS

containerField

USE ☐ ILS☐ geometry

colorIndex

colorPerVertex ☒


coordIndex

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
0 -1

OK

Cancel

Help

 IndexedLineSet	<p>IndexedLineSet is a geometry node that can contain a Color node and a Coordinate node. Color values or a Material emissiveColor is used to draw lines and points. Lines are not lit, are not texture-mapped, and do not participate in collision detection.</p> <p>Hint: use a different color (or emissiveColor) than the background color.</p> <p>Hint: if rendering Coordinate points originally defined for an IndexedFaceSet, index values may need to repeat each initial vertex to close each polygon outline. Step-wise colors or linear interpolation of colors can be used as a good scientific visualization technique to map arbitrary function values to a color map.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
coordIndex	<p>[coordIndex: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</p> <p>coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets, use -1 to separate indices for each polyline.</p> <p>Hint: if rendering Coordinate points originally defined for an IndexedFaceSet, index values may need to repeat initial each initial vertex to close the polygons.</p>
colorPerVertex	<p>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</p> <p>Whether Color node is applied per vertex (true) or per polyline (false).</p>
colorIndex	<p>[colorIndex: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</p> <p>colorIndex indices provide order in which colors are applied.</p> <p>Hint: if rendering Coordinate points originally defined for an IndexedFaceSet, index values may need to repeat initial each initial vertex to close the polygons.</p>
set_coordIndex	<p>[set_coordIndex: accessType inputOnly, type MInt32 CDATA #FIXED ""]</p> <p>coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.</p>
set_colorIndex	<p>[set_colorIndex: accessType initializeOnly, type MInt32 CDATA #FIXED ""]</p> <p>colorIndex indices provide order in which colors are applied.</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

LineSet node

Similar to IndexedLineSet

- Contain 0 or 1 Coordinate/CoordinateDouble
- Material *emissiveColor* or Color/ColorRGBA

Rather than using coordIndex and colorIndex,
LineSet has *vertexCount* field

- *vertexCount* MFInt32 array of integers defines number of sequential points used in each polyline
- No -1 sentinel values needed
- Color and Coordinate values used in defined order
- Somewhat more compact than using indices

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/spe
<head>
  <meta content='LineSet.x3d' name='title' />
  <meta content='The way points and path of the animated shark Lucy traversing the tank.' name='description' />
  <meta content='Tim McLean' name='creator' />
  <meta content='Don Brutzman' name='translator' />
  <meta content='June 1998' name='created' />
  <meta content='20 December 2002, 11 March 2006' name='modified' />
  <meta content='http://web.nps.navy.mil/~brutzman/kelp' name='reference' />
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/SharkLucyLocale.x3d' name='reference' />
  <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/LineSet.x3d' name='identifier' />
  <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
  <meta content='../license.html' name='license' />
</head>
<Scene>
  <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapt
  <ProtoInstance name='WhereAmI' />
  <Background skyColor='1 1 1' />
  <Viewpoint description='Book View' orientation='0.939 0.335 0.075 -0.57' position='-0.89 1.91 9.26' />
  <Transform DEF='_0' translation='0.0 -8.0 -1.0'>
    <Inline url='"SharkLucy.wrl"' "http://X3dGraphics.com/examples/
  <Group>
    <TimeSensor DEF='SHARK1_CLOCK' cycleInterval='220.0' loop='t
    <PositionInterpolator DEF='SHARK1_POSITION' key='0.0 0.048 0
    <OrientationInterpolator DEF='SHARK1_ORIENTATION' key='0.0 0
  </Group>
</Transform>
<Transform translation='0 0 0'>
  <Shape>
    <Appearance>
      <Material emissiveColor='0 0 1' />
    </Appearance>
    <LineSet vertexCount='19'>
      <Coordinate point='0.0 -7.0 -1.0 -1.75 -7.0 -0.5 -4.0 -7.0 0.5 -5.0 -6.5 1.5 -5.5 -6.25 0.75 -5.25 -5.5 -2.25 -4.25 -5.0 -3.25 -2.75 -4.5
    </LineSet>
  </Shape>
  <Shape>
    <Appearance>
      <Material emissiveColor='1 0 0' />
    </Appearance>
    <PointSet>
      <Coordinate point='0.0 -7.0 -1.0 -1.75 -7.0 -0.5 -4.0 -7.0 0.5 -5.0 -6.5 1.5 -5.5 -6.25 0.75 -5.25 -5.5 -2.25 -4.25 -5.0 -3.25 -2.75 -4.5
    </PointSet>
  </Shape>

```

Edit LineSet

DEF ☐

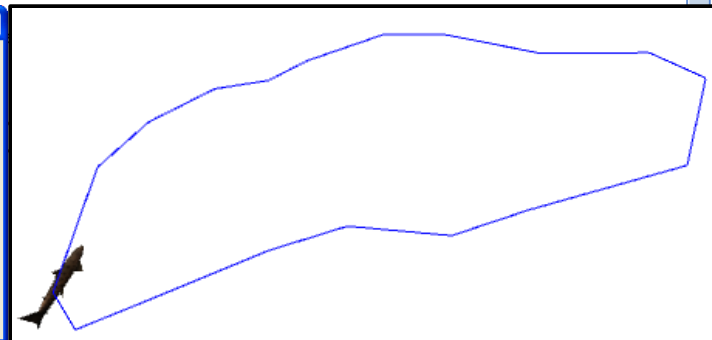
USE ☐


containerField

☐ geometry

vertexCount

OK Cancel Help



 LineSet	<p>LineSet is a geometry node that can contain a Color node and a Coordinate node. Color values or a Material emissiveColor is used to draw lines and points. Lines are not lit, are not texture-mapped, and do not participate in collision detection.</p> <p>Hint: use a different color (or emissiveColor) than the background color. Linear interpolation of colors can be used as a good scientific visualization technique to map arbitrary function values to a color map.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
vertexCount	<p>[vertexCount: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>[2,infinity) vertexCount describes how many vertices are used in each polyline from Coordinate field. Coordinates are assigned to each line by taking vertexCount[n] vertices from Coordinate field.</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

IndexedFaceSet node 1

IndexedFaceSet creates a set of polygons (faces)

- Contains Coordinate node for *point* data
- Can be discontinuous or share points repeatedly

Color can be set in one of two ways

- Uniformly via sibling Material fields
- Individually via contained Color/ColorRGBA node; applied either by individual points, or by each polygon, as determined by *colorPerVertex*

IndexedFaceSet node 2

Many fields and features apply

- *ccw, convex, solid, creaseAngle* as before
- *colorPerVertex, normalPerVertex* as before
- *coordIndex, colorIndex, normalIndex* as before
- *texCoordIndex* applies texture coordinates to map texture images to individual geometry points

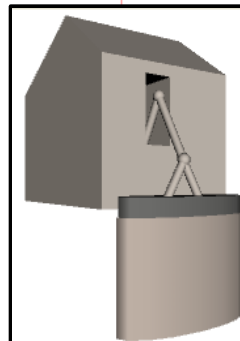
Contained nodes (0 or 1 of each)

- Color/ColorRGBA
- Coordinate/CoordinateDouble
- Normal, TextureCoordinate

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d-3.1.xsd'>
  <head>
    <meta content='IndexedFaceSet.x3d' name='title' />
    <meta content='Positive-displacement cylinder pump house built using IndexedFaceSet nodes.' name='description' />
    <meta content='Todd Gagnon and Mark A. Boyd' name='authors' />
    <meta content='Xeena VRML importer' name='translator' />
    <meta content='8 June 1998' name='created' />
    <meta content='20 December 2002' name='imported' />
    <meta content='7 March 2006' name='modified' />
    <meta content='KelpTank.x3d' name='reference' />
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/PumpHouse.x3d' name='reference' />
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/IndexedFaceSet.x3d' name='identifier' />
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
    <meta content='Vrm197ToX3dNist, http://ovrt.nist.gov/v2_x3d.html' name='generator' />
    <meta content='../license.html' name='license' />
    <meta content='This is not really static because the PositionInterpolators contain an initial offset of the piston and rocker arm.' name='comment' />
  </head>
  <Scene>
    <ExternProtoDeclare name='WhereAmI' url='../Chapter14-Prototypes/WhereAmI.x3d#WhereAmI' "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter14-Prototypes/WhereAmI.x3d#WhereAmI" />
    <ProtoInstance name='WhereAmI' />
    <Background skyColor='1 1 1' />
    <Viewpoint description='Book View' orientation='0 -1 0 0.53' position='-2.28 0.29 4.06' />
    <Group>
      <Shape>
        <Appearance DEF='pumpHouse'>
          <Material diffuseColor='0.82 0.78 0.74' />
        </Appearance>
        <IndexedFaceSet DEF='IFS' coordIndex='0 1 5 4 -1 5 1 2 6 -1 6 2 3 7 -1 3 0 4 7 -1 1 12' />
        <Coordinate point='0.0 0.0 0.0 2.0 0.0 0.0 2.0 1.75 0.0 0.0 1.75 0.0 0.625 0.75 0.0 1' />
      </IndexedFaceSet>
    </Shape>
    <Transform scale='0.91 0.6 0.3' translation='0.8 -0.65 0.5'>
      <Shape>
        <Appearance>
          <Material diffuseColor='0.749 0.694 0.651' />
        </Appearance>
        <Appearance>
          <Cylinder bottom='false' top='false' />
        </Appearance>
      </Shape>
    </Transform>
  </Group>
  <Group>
    <Transform scale='0.5 0.5 0.5' translation='1.0 1.1 -1.5'>
      <Transform DEF='PISTON'>
        <Transform scale='1.8 1.2 0.6' translation='0.0 -0.2 0.0'>

```



Edit IndexedFaceSet

DEF ☐ IFS

USE ☐ IFS

containerField

☐ geometry

ccw ☒

convex ☒

solid ☒

creaseAngle

colorPerVertex ☒

normalPerVertex ☒


colorIndex

coordIndex

normalIndex

texCoordIndex

OK Cancel Help

 IndexedFaceSet	<p>IndexedFaceSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
coordIndex	<p>[coordIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.</p>
ccw	<p>[ccw: accessType initializeOnly, type SFBBool (true false) "true"]</p> <p>ccw = counterclockwise: ordering of vertex coordinates orientation.</p> <p>Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.</p>
convex	<p>[convex: accessType initializeOnly, type SFBBool (true false) "true"]</p> <p>Whether all polygons in a shape are convex (true), or possibly concave (false) A convex polygon is planar, does not intersect itself, and has all interior angles < 180 degrees.</p> <p>Interchange profile hint: only convex=true IndexedFaceSets are supported.</p> <p>Warning: concave geometry may be invisible default convex=true.</p>
solid	<p>[solid: accessType initializeOnly, type SFBBool (true false) "true"]</p> <p>Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off).</p> <p>Warning: default value true can completely hide geometry if viewed from wrong side!</p>
creaseAngle	<p>[creaseAngle: accessType initializeOnly, type SFFloat CDATA "0"]</p> <p>[0..infinity) creaseAngle defines angle (in radians) for determining whether adjacent polygons are drawn with sharp edges or smooth shading. If angle between normals of two adjacent polygons is less than creaseAngle, smooth shading is rendered across the shared line segment.</p> <p>Interchange profile hint: only 0 and $\hat{1}$ radians supported.</p> <p>Hint: creaseAngle=0 means render all edges sharply, creaseAngle=3.14 means render all edges smoothly.</p>
colorPerVertex	<p>[colorPerVertex: accessType initializeOnly, type SFBBool (true false) "true"]</p> <p>Whether Color node is applied per vertex (true) or per polygon (false).</p>
colorIndex	<p>[colorIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>colorIndex indices provide order in which colors are applied.</p>
normalPerVertex	<p>[normalPerVertex: accessType initializeOnly, type SFBBool (true false) "true"]</p> <p>Whether Normal node is applied per vertex (true) or per polygon (false).</p>
normalIndex	<p>[normalIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>Interchange profile hint: this field may be ignored.</p>

texCoordIndex	<p>[texCoordIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>List of texture-coordinate indices mapping attached texture to corresponding coordinates.</p> <p>Hint: use a tool!</p>
set_coordIndex	<p>[set_coordIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""]</p> <p>coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.</p>
set_colorIndex	<p>[set_colorIndex: accessType initializeOnly, type MFInt32 CDATA #FIXED ""]</p> <p>colorIndex indices provide order in which colors are applied.</p>
set_normalIndex	<p>[set_normalIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""]</p> <p>Interchange profile hint: this field may be ignored.</p>
set_texCoordIndex	<p>[set_texCoordIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""]</p> <p>List of texture-coordinate indices mapping attached texture to corresponding coordinates.</p> <p>Hint: use a tool!</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

ElevationGrid node

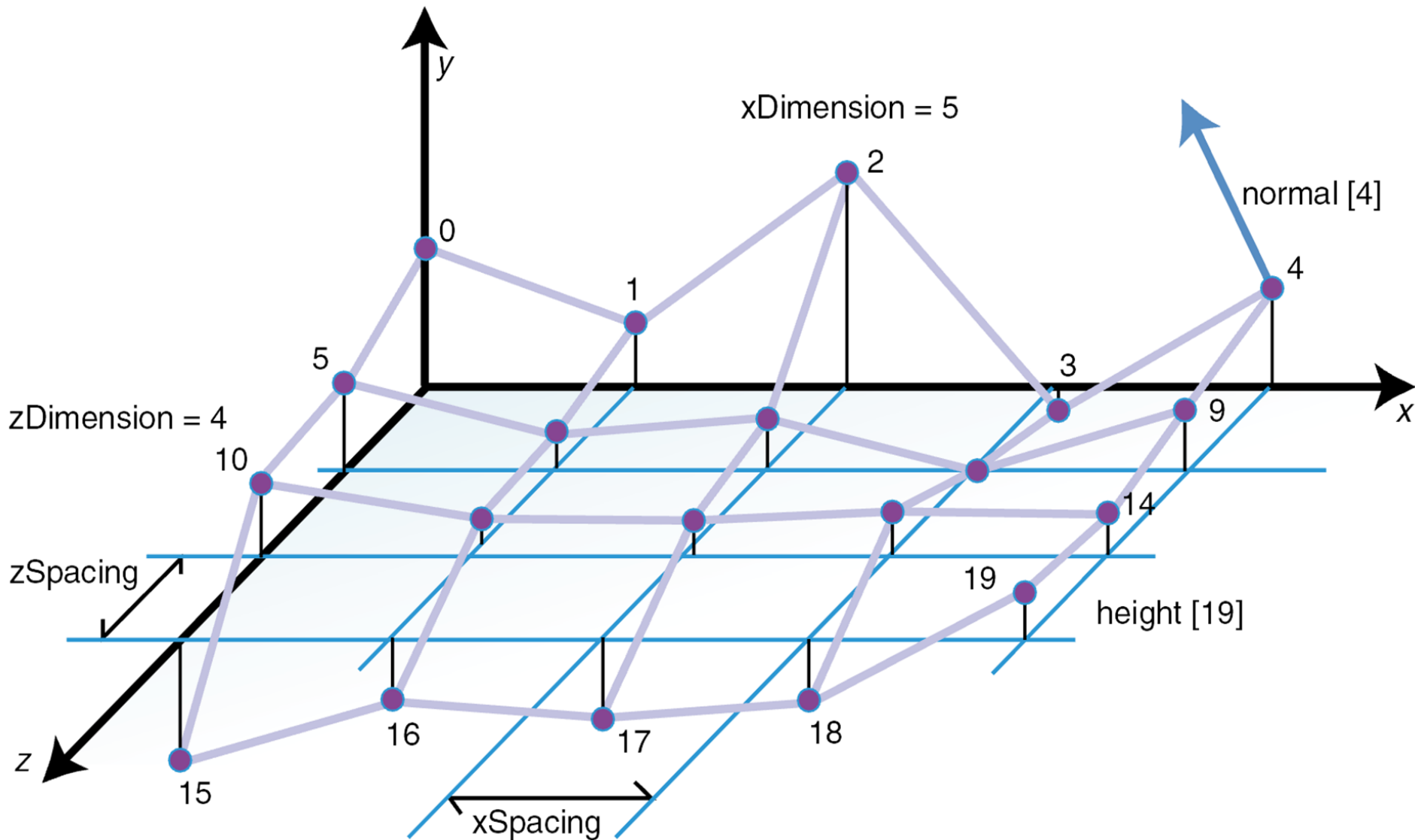
ElevationGrid takes a rectangular array of floats and converts *height* array into post values above (or below) baseline $y=0$ ground plane

- *xSpacing*, *zSpacing* are lengths in meters
- *xDimension*, *zDimension* are row, column sizes
- *height* MFFloat array size $xDimension \cdot zDimension$
- *ccw*, *solid* as before
- *colorPerVertex*, *normalPerVertex* as before

Contained nodes (0 or 1 of each)

- Color/ColorRGBA, Normal, TextureCoordinate

ElevationGrid indexing of *height* array

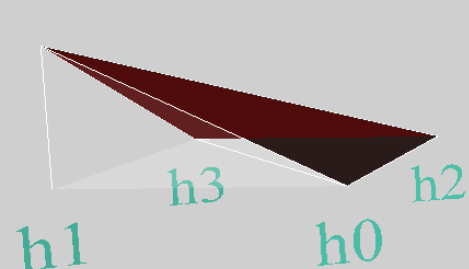


ElevationGrid node

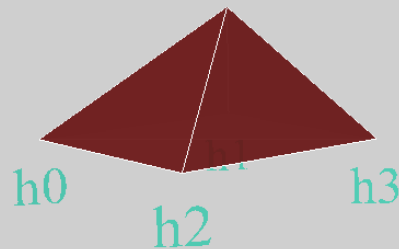
Alternate forms of tessellation are possible for nonplanar ElevationGrid quadrilaterals

- Almost all ElevationGrid quads are nonplanar, otherwise the geometry is flat
- Leftmost two figures show different views of grid
- Rightmost two figures show different tessellations
- Usually not a problem for large, fine-scaled grid

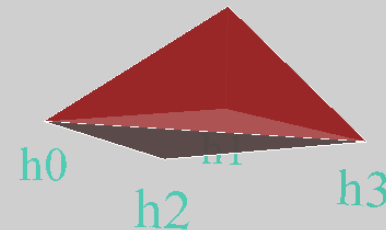
Browser
ElevationGrid



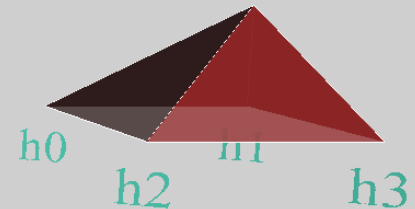
Browser
ElevationGrid



Center Diagonal
Tessellation




Cross Diagonal
Tessellation



ElevationGrid node X3D-Edit

[example TBD]

 ElevationGrid	<p>ElevationGrid is a geometry node. ElevationGrid is a rectangular grid of varying height above a flat surface. ElevationGrid can contain Color, Normal and TextureCoordinate nodes.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
xDimension	<p>[xDimension: accessType initializeOnly, type SInt32 CDATA "0"]</p> <p>Number of grid-array elements along X direction.</p>
zDimension	<p>[zDimension: accessType initializeOnly, type SInt32 CDATA "0"]</p> <p>Number of grid-array elements along Z direction.</p>
xSpacing	<p>[xSpacing: accessType initializeOnly, type SFloat CDATA "1.0"]</p> <p>Meters distance between grid-array vertices along X direction.</p> <p>Hint: total horizontal x-axis distance equals (xDimension-1) * xSpacing.</p>
zSpacing	<p>[zSpacing: accessType initializeOnly, type SFloat CDATA "1.0"]</p> <p>Meters distance between grid-array vertices along Z direction.</p> <p>Hint: total vertical z-axis distance equals (zDimension-1) * zSpacing.</p>
height	<p>[height: accessType initializeOnly, type MFloat CDATA #IMPLIED]</p> <p>Grid array of height vertices along upward Y direction, with xDimension rows and zDimension columns.</p>
set_height	<p>[set_height: accessType inputOnly, type MFloat CDATA #FIXED ""]</p> <p>Grid array of height vertices along upward Y direction, with xDimension rows and zDimension columns.</p>
ccw	<p>[ccw: accessType initializeOnly, type SBool (true false) "true"]</p> <p>ccw = counterclockwise: ordering of vertex coordinates orientation.</p> <p>Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.</p>
creaseAngle	<p>[creaseAngle: accessType initializeOnly, type SFloat CDATA "0"]</p> <p>[0..infinity) creaseAngle defines angle (in radians) for determining whether adjacent polygons are drawn with sharp edges or smooth shading. If angle between normals of two adjacent polygons is less than creaseAngle, smooth shading is rendered across the shared line segment.</p> <p>Hint: creaseAngle=0 means render all edges sharply, creaseAngle=3.14 means render all edges smoothly.</p>
solid	<p>[solid: accessType initializeOnly, type SBool (true false) "true"]</p> <p>Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off).</p> <p>Warning: default value true can completely hide geometry if viewed from wrong side!</p>
colorPerVertex	<p>[colorPerVertex: accessType initializeOnly, type SBool (true false) "true"]</p> <p>Whether Color node is applied per vertex (true) or per quadrilateral (false).</p>
normalPerVertex	<p>[normalPerVertex: accessType initializeOnly, type SBool (true false) "true"]</p> <p>Whether Normal node is applied per vertex (true) or per quadrilateral (false).</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape.</p> <p>containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

Extrusion node 1

Extrusion begins with a planar *crossSection* outline, then stretches (extrudes) it along a *spine* polyline

- *crossSection* is MFVec2f array of 2-tuple floating-point 2D coordinate pairs creating an outline
- *spine* is MFVec3f array of 3-tuple floating-point 3D coordinates creating a polyline

Extrusion is a bit tricky to master, but provides a great way to create sophisticated geometry with little effort

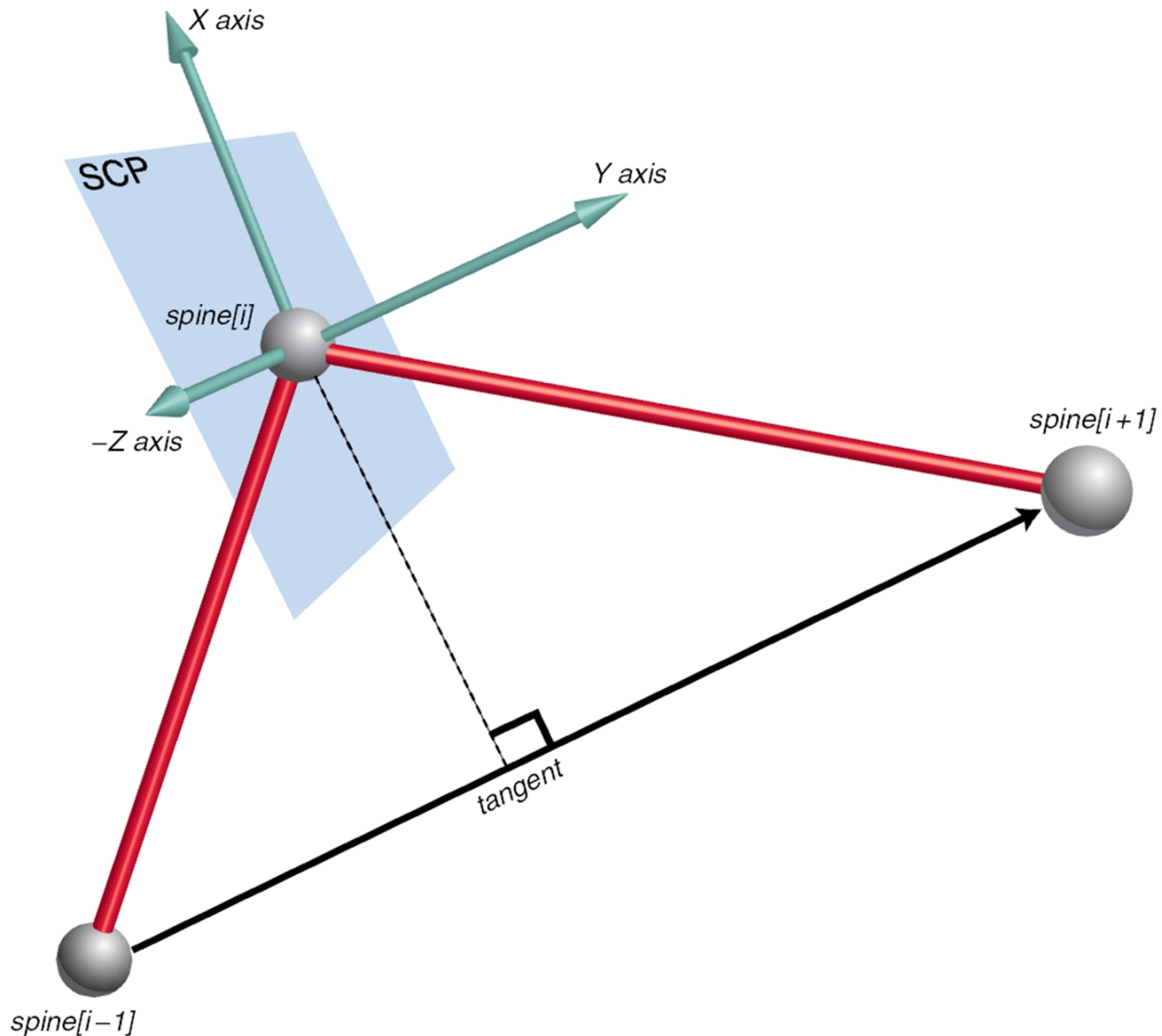
Extrusion node 2

Spine-aligned cross-section plane (SCP) refers to each individual copy of the cross section, each of which appear about each *spine* point

- Extrusion outer hull simply connects corresponding points on these cross-section outlines
- If the outline of the Extrusion is degenerate or ill defined, then the polygons making up Extrusion outline are similarly confounded

Drawing simple outlines of *crossSection* on graph paper is great way to keep things straight

Extrusion spine-aligned cross-section plane (SCP)



Extrusion node 2

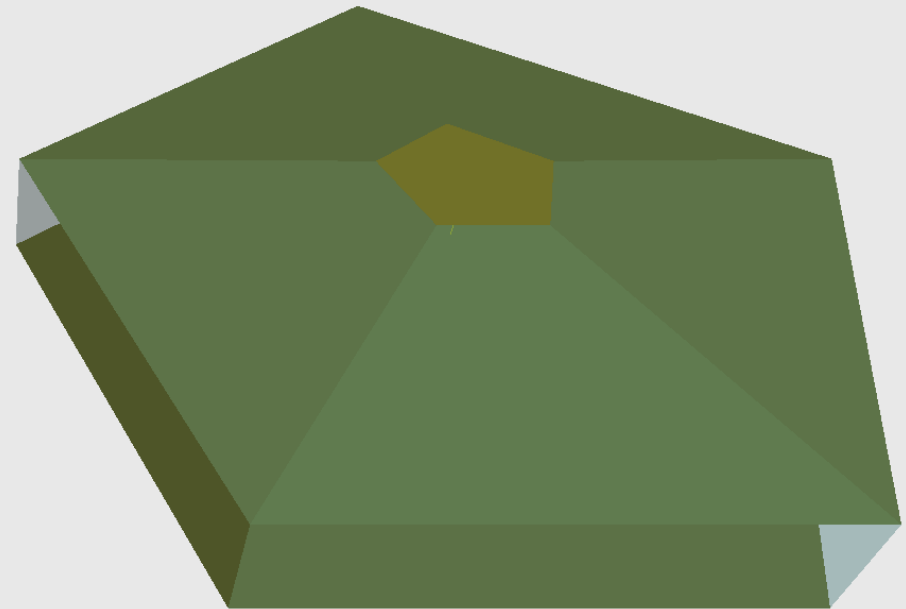
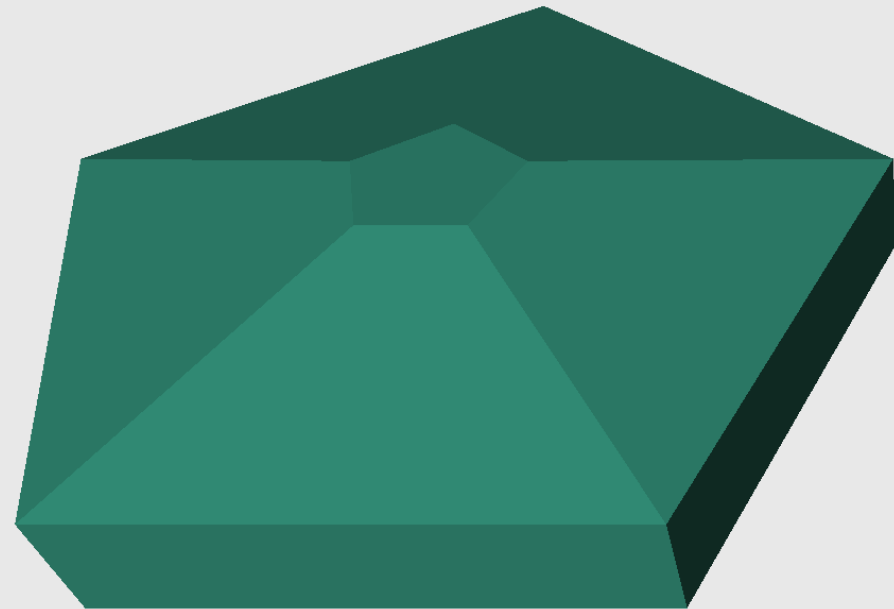
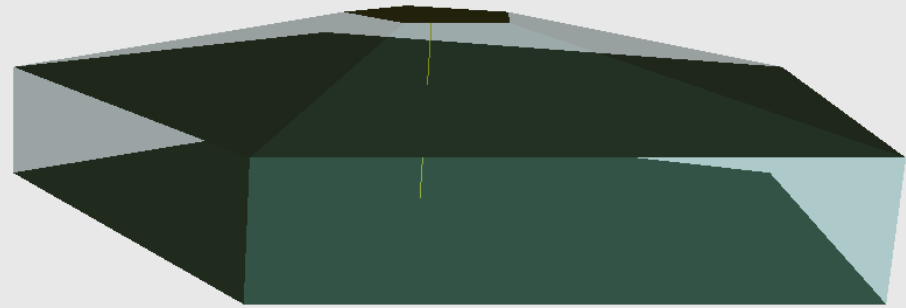
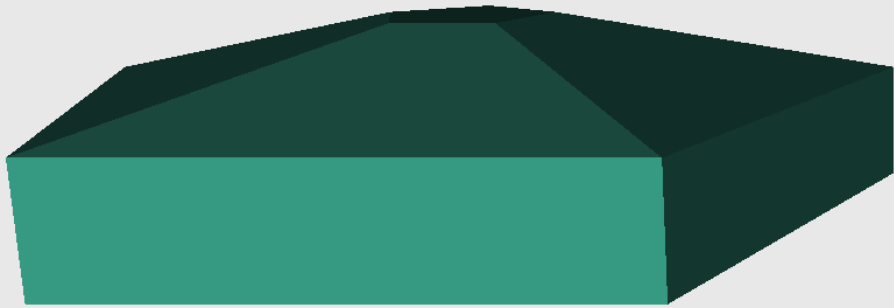
Further modifications include *scale*, *orientation* to modify each cross-section about SCP center

- *scale* is MFVec2f array of 2-tuple floating-point pairs to scale the local spine-aligned crossSection plane
- *orientation* is MFRotation array to rotate
- Single value affects all simultaneously, array affects each repeated cross-section individually

Other fields are common

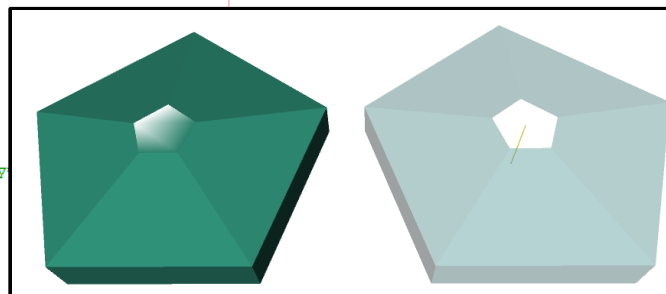
- *ccw*, *convex*, *solid*, *creaseAngle* as before
- *beginCap*, *endCap* are SFBool values to close ends

Extrusion cross-section example: pentagon





```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d-3.1.xsd'>
  <head>
    <meta content='ExtrusionPentagon.x3d' name='title' />
    <meta content='Simple regular pentagon extruded vertically.' name='description' />
    <meta content='Don Brutzman' name='creator' />
    <meta content='Naval Postgraduate School' name='organization' />
    <meta content='3 September 2005' name='created' />
    <meta content='3 September 2005' name='modified' />
    <meta content='Extrusion pentagon' name='subject' />
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionPentagon.x3d' name='identifier' />
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
    <meta content='../license.html' name='license' />
  </head>
  <Scene>
    <NavigationInfo DEF='ExamineMode' type='EXAMINE' "ANY" />
    <NavigationInfo DEF='FlyPanNavigation' type='FLY' "ANY" />
    <Background skyColor='1 1 1' />
    <Viewpoint description='ExtrusionPentagon' orientation='-1 0 0 0.2' position='0 3.17 12.85' />
    <Viewpoint description='Oblique view from above' orientation='-1 0.015 -0.006 0.78' position='0.25 8.98 8.99' />
    <Viewpoint description='Overhead view' orientation='1 0 0 -1.57' position='0 13.15 0' />
    <Background groundColor='0.9 0.9 0.9' skyColor='0.9 0.9 0.9' />
    <Transform translation='-4 0 0'>
      <Shape>
        <!-- be sure to order cross-section points so that normal is upward -->
        <Extrusion crossSection='-3.5 -1 -2.1 2.9 2.2 2.9 3.6 -1 0 -3.5 -3.5 -1' scale='1 1 1 1 0.2 0.2' solid='true' spine='0 0 0 0 1 0 0 1.6 0' />
        <Appearance>
          <Material diffuseColor='0.2 0.6 0.5' />
        </Appearance>
      </Shape>
    </Transform>
    <!-- Utilize ExtrusionCrossSection ProtoInstance as a Shape node (with containerField="children")
        rather than an Extrusion node (with containerField="geometry"). -->
    <ExternProtoDeclare>
      <Transform translation='4 0 0'>
        <ProtoInstance DEF='PentagonExtrusion' name='ExtrusionCrossSection'>
          <fieldValue name='name' value='PentagonExtrusion' />
          <fieldValue name='crossSection' value='-3.5 -1 -2.1 2.9 2.2 2.9 3.6 -1 0 -3.5 -3.5 -1' />
          <fieldValue name='spine' value='0 0 0 0 1 0 0 1.6 0' />
          <fieldValue name='scale' value='1 1 1 1 0.2 0.2' />
          <fieldValue name='lineColor' value='0.7 0.7 0' />
          <fieldValue name='ccw' value='true' />
          <fieldValue name='crossSectionMaterial'>
            <Material diffuseColor='0.6 0.6 0.2' />
          </fieldValue>
          <fieldValue name='extrusionMaterial'>
            <Material diffuseColor='0.3 0.6 0.6' transparency='0.5' />
          </fieldValue>
          <fieldValue name='traceEnabled' value='false' />
        </ProtoInstance>
      </Transform>
    </Scene>
  </X3D>
```



Edit Extrusion

DEF ☒ USE ☐

containerField

ccw ☒ convex ☒ beginCap ☒ endCap ☒ solid ☒

creaseAngle

crossSection

spine

scale

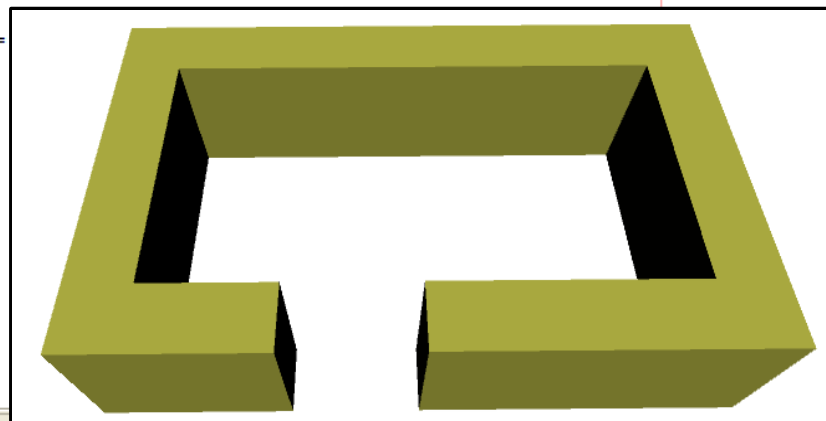
orientation

OK Cancel Help





```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance'
      xsd:noNamespaceSchemaLocation='http://www.web3d.org/specifications/x3d-3.1.xsd'>
  <head>
    <meta content='ExtrusionRoomWalls.x3d' name='title' />
    <meta content='Wall definition for a room, defined as a cross section and extruded vertically.' name='description' />
    <meta content='Don Brutzman' name='creator' />
    <meta content='Naval Postgraduate School' name='organization' />
    <meta content='3 September 2005' name='created' />
    <meta content='3 September 2005' name='modified' />
    <meta content='Extrusion pentagon' name='subject' />
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionRoomWalls.x3d'
          name='identifier' />
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator' />
    <meta content='../license.html' name='license' />
  </head>
  <Scene>
    <Viewpoint description='ExtrusionRoomWalls' orientation='1 0 0 -0.2' position='5 4 15' />
    <Viewpoint description='Oblique view from above' orientation='1 0 0 -0.78' position='5 10 13' />
    <Viewpoint description='Overhead view' orientation='1 0 0 -1.57' position='5 12 3' />
    <Background skyColor='1 1 1' />
    <Transform>
      <Shape>
        <!-- be sure to order cross-section points so that normal is upward -->
        <Extrusion crossSection='0 0 0 6 3 6 3 5 1 5 1 1 9 1 9 5 5 5 5 6 10 6 10 0'
                  convex='false' solid='true' spine='0 0 0 0 2.5 0' />
        <Appearance>
          <Material diffuseColor=
        </Appearance>
      </Shape>
    </Transform>
  </Scene>
</X3D>
```



Edit Extrusion

DEF ☐ USE ☐

containerField ☐ geometry

ccw ☒

convex ☒

beginCap ☒

endCap ☒

solid ☒

creaseAngle

crossSection

spine

scale

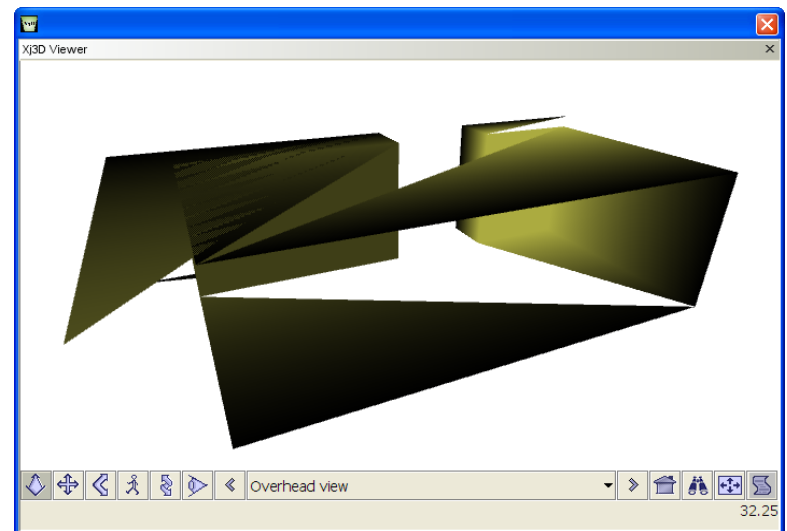
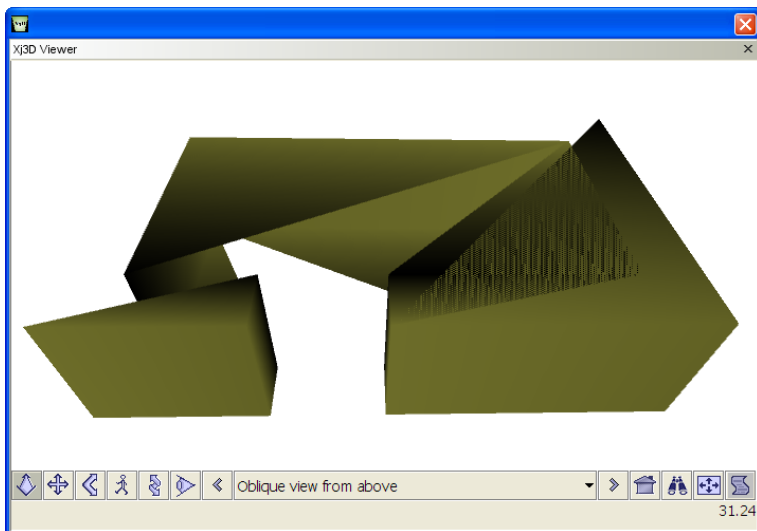
orientation

OK Cancel Help

Concave geometry effects

Concave geometry with *convex*='true'
can lead to confused geometry results

- Nonsensical polygons
- Aliasing (tearing) of coplanar polygons
- To correct: set *convex*='false'



Debugging Extrusion problems 1

Unlike most other nodes, ill-defined geometry is possible with Extrusion. Things to check:

- Verify proper *crossSection*, *spine*, *scale*, *orientation* array values and lengths
- Set *convex*='false' if geometry might be concave
- Set *solid*='false' to render inside and outside, eliminating 'invisible geometry' when viewed from behind or inside the exterior hull
- Set *ccw*='false' if crossSection might be defined in clockwise direction

Debugging Extrusion problems 2

Counting checks

- Length of 2-tuple *scale* array must be 0, 1, or match length of 3-tuple *spine* array
- Length of 4-tuple *orientation* array must be 0, 1, or match length of 3-tuple *spine* array
- Values in *scale* and *crossSection* arrays must be multiple of 2 (MFVec2f)
- Values in *spine* array must be multiple of 3 (MFVec3f)
- Values in *orientation* array must be multiple of 4 (MFRotation)

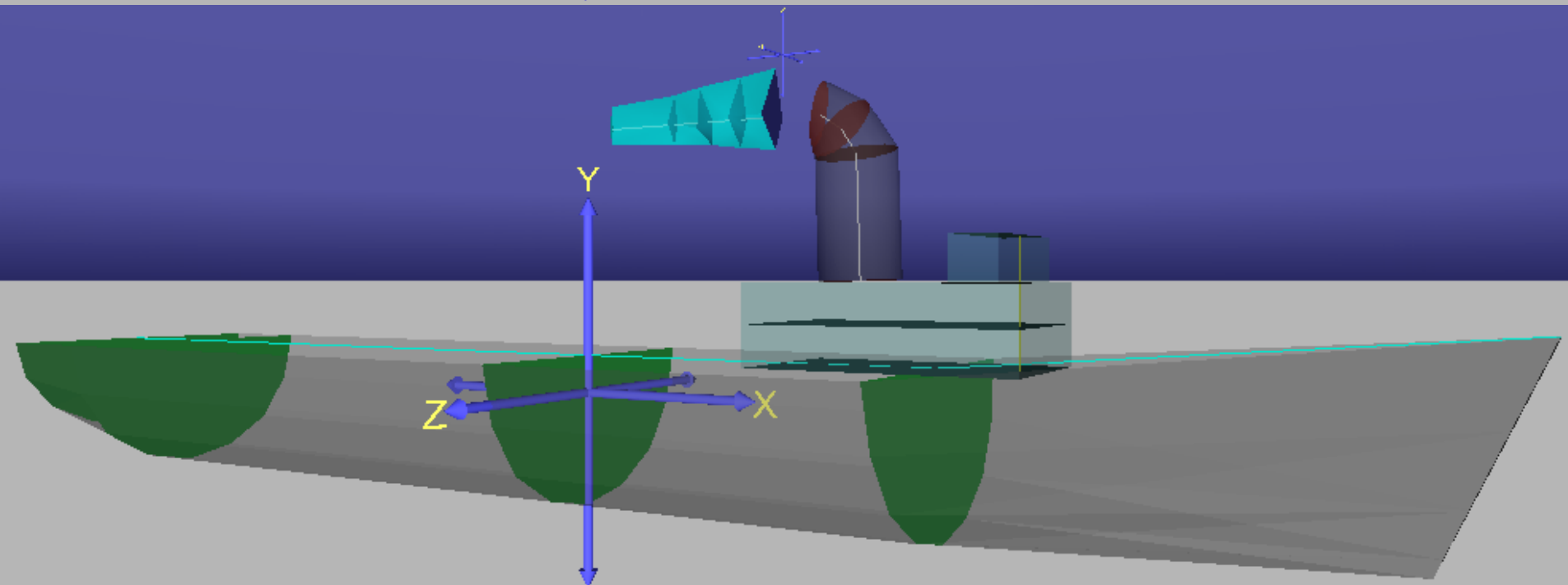
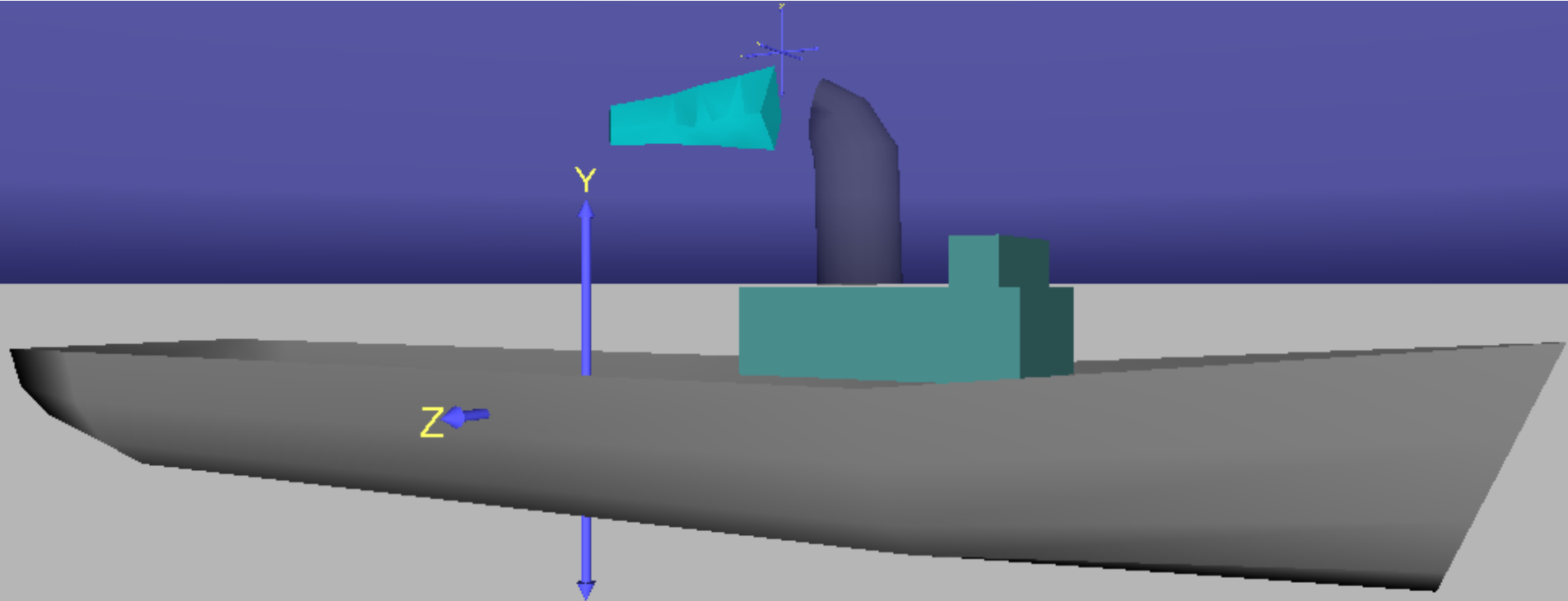
ExtrusionCrossSection prototype

The preceding ExtrusionPentagon.x3d example scene contains new construct: Prototype node

- ExternProtoDeclare refers to external prototype url and defines field signatures
- ProtoInstance creates an instance of new node
- fieldValue definitions provide parameter values, in this case the same values as Extrusion of interest

Result is a specially computed Extrusion showing *crossSection* planes, *spine*, transparent sides

- Can provide helpful insight and debugging support




```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" "http://www.web3d.org/specifications/x3d-3.0.dtd">
<X3D profile='Immersive' version='3.0' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/spe
<head>
  <meta content='ExtrusionCrossSectionExampleShip.x3d' name='title' />
  <meta content='Don Brutzman' name='creator' />
  <meta content='20 December 1999' name='created' />
  <meta content='10 July 2006' name='modified' />
  <meta content='Ship model as example use of ExtrusionCrossSectionPrototype for drawing several different Extrusion spines and cross sections.' name='description' />
  <meta content='ExtrusionCrossSectionExampleShip.png' name='image' />
  <meta content='Utilize ExtrusionCrossSection ProtoInstance as a Shape node (with containerField="children") rather than an Extrusion node (with containerField="geometry"). -->
  <meta content='ExtrusionExampleShip.x3d' name='reference' />
  <meta content='VRML 97 Specification, 6.18 Extrusion' name='reference' />
  <meta content='http://www.web3d.org/technicalinfo/specifications/vrml97/part1/nodesRef.html#Extrusion' name='reference' />
  <meta content='VRML 97 Specification, Figure 6.6' name='image' />
  <meta content='http://www.web3d.org/technicalinfo/specifications/vrml97/part1/Images/Extrusion.gif' name='image' />
  <meta content='http://www.web3d.org/x3d/content/examples/Basic/ExtrusionCrossSectionExampleShip.x3d' name='identifier' />
  <meta content='X3D-Edit, http://www.web3d.org/x3d/content/README.X3D-Edit.html' name='generator' />
  <meta content='../..//license.html' name='license' />
</head>
<Scene>
  <!-- Utilize ExtrusionCrossSection ProtoInstance as a Shape node (with containerField="children") rather than an Extrusion node
    (with containerField="geometry"). -->
  <ExternProtoDeclare>
    <!-- ===== -->
    <!-- Example instance -->
    <Viewpoint description='ship hull' position='0 0 40' />
    <Viewpoint description='stack' position='8 5 20' />
    <Viewpoint description='smoke extrusion start' position='6.8 7 10' />
    <Viewpoint description='forward starboard quarter' orientation='0 1 0 0.5' position='20 3 25' />
    <Transform rotation='0 1 0 1.57' translation='15 30 0'>
      <Viewpoint description='touch smoke to animate' orientation='1 0 0 -1' position='0 0 0' />
    </Transform>
    <NavigationInfo speed='4' />
    <Background groundColor='0.7 0.7 0.7' skyAngle='0.05 1.5 1.59' skyColor='0.6 0.6 0.4 0.4 0.4 0.7 0.3 0.3 0.6 0.1 0.1 0.3' />
    <!-- ===== -->
    <Transform DEF='SmokePosition' translation='5.8 7 0'>
      <!-- Smoke shows that crossSection, spine, scale and orientation can be animated -->
      <ProtoInstance DEF='Smoke' name='ExtrusionCrossSection'>
        <fieldValue name='name' value='Smoke' />
        <fieldValue name='crossSection' value='0 1 -1 0 0 -0.5 1.5 0 0 1' />
        <fieldValue name='spine' value='0 0 0 -1 0 0 -2 0 0 -3 0 0 -5 0 0' />
        <fieldValue name='scale' value='0.8 0.7 0.7 0.6 0.6 0.5 0.5 0.3 0.4 0.2' />
        <fieldValue name='orientation' value='0 1 0 0 0 1 0 0 0 1 0 -0.4 0 1 0 0 1 0 0 0.4' />
        <fieldValue name='creaseAngle' value='1' />
        <fieldValue name='solid' value='false' />
      </ProtoInstance>
    </Transform>
  </Scene>

```

ExtrusionCrossSectionExampleShip.x3d part 1




```

<fieldValue name='lineColor' value='1 1 1'>
<fieldValue name='crossSectionMaterial'>
  <Material diffuseColor='0.2 0.2 0.6'>
</fieldValue>
<fieldValue name='extrusionMaterial'>
  <Material diffuseColor='0.0 0.9 0.9' transparency='0.3'>
</fieldValue>
<fieldValue name='traceEnabled' value='false'>
</ProtoInstance>
<TouchSensor DEF='TouchSmoke' description='click smoke to animate'>
<TimeSensor DEF='SmokeClock' cycleInterval='30' startTime='-1'>
<CoordinateInterpolator DEF='SmokeSpineInterpolator' key='0 0.2 0.5 0.8 1' keyValue='0 0 0 -1 0 0 -2 0 0 -3 0 0 -5 0 0 0 0 0 -2 0 0 -4 -0.5 0'>
<ROUTE fromField='touchTime' fromNode='TouchSmoke' toField='startTime' toNode='SmokeClock'>
<ROUTE fromField='fraction_changed' fromNode='SmokeClock' toField='set_fraction' toNode='SmokeSpineInterpolator'>
<ROUTE fromField='value_changed' fromNode='SmokeSpineInterpolator' toField='set_spine' toNode='Smoke'>
</Transform>
<Transform DEF='SuperstructurePosition' translation='12 1 0'>
  <ProtoInstance DEF='Superstructure' name='ExtrusionCrossSection'>
    <fieldValue name='name' value='Superstructure'>
    <fieldValue name='crossSection' value='0.1 1 0.1 -1 -1 -1 -1 1 0.1 1'>
    <fieldValue name='spine' value='0 0 0 1 0 0 1.95 0 0 1.96 0 0 3 0'>
    <fieldValue name='scale' value='6 2 6 2 6 2 1 1.8 1 1.8'>
    <fieldValue name='lineColor' value='0.7 0.7 0'>
    <fieldValue name='ccw' value='true'>
    <fieldValue name='crossSectionMaterial'>
      <Material diffuseColor='0.6 0.6 0.2'>
    </fieldValue>
    <fieldValue name='extrusionMaterial'>
      <Material diffuseColor='0.3 0.6 0.6' transparency='0.6'>
    </fieldValue>
    <fieldValue name='traceEnabled' value='false'>
  </ProtoInstance>
</Transform>
<Transform DEF='StackPosition' translation='8 1 0'>
  <ProtoInstance DEF='Stack' name='ExtrusionCrossSection'>
    <fieldValue name='name' value='Stack'>
    <fieldValue name='crossSection' value='0 1 0.38 0.92 0.71 0.71 0.92 0.38 1 0 0.92 -0.38 0.71 -0.71 0.38 -0.92 0 -1 -0.38 -0.92 -0.71 -0.71 -'>
    <fieldValue name='spine' value='0.1 2 0 0 5 0 -0.4 5.6 0 -1 6 0'>
    <fieldValue name='scale' value='1 1 1 1 0.9 0.85 0.8 0.4'>
    <fieldValue name='orientation' value='0 1 0 0 0 1 0 0 0 1 0 0 0 1 0.4'>
    <fieldValue name='lineColor' value='1 1 1'>
    <fieldValue name='creaseAngle' value='1'>
    <fieldValue name='crossSectionMaterial'>
      <Material diffuseColor='0.8 0.2 0.2' emissiveColor='0.1 0 0' transparency='0.1'>
    </fieldValue>
    <fieldValue name='extrusionMaterial'>

```

ExtrusionCrossSectionExampleShip.x3d part 2

 Extrusion	<p>Extrusion is a geometry node stretching a 2D cross section along a 3D-spine path in the local coordinate system Scaling/rotating cross-sections can produce a variety of shapes.</p> <p>Hint: insert a Shape node before adding geometry or Appearance.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring <code>_all_</code> other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
spine	<p>[spine: accessType initializeOnly, type MFVec3f CDATA "0 0 0, 0 1 0"]</p> <p>spine is a list of 3D points for a piecewise-linear curve forming a series of connected vertices, open or closed. This is the path along which the crossSection is extruded.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p>
crossSection	<p>[crossSection: accessType initializeOnly, type MFVec2f CDATA "1 1, 1 -1, -1 -1, -1 1, 1 1"]</p> <p>An ordered set of 2D points drawing a piecewise-linear curve and forming a planar series of connected vertices. This provides a silhouette of the outer surface.</p> <p>Warning: match clockwise/counterclockwise or impossible/inverted geometry can result!</p>
scale	<p>[scale: accessType initializeOnly, type MFVec2f CDATA "1 1"]</p> <p>(0..infinity) scale is a list of 2D-scale parameters applied at each spine-aligned cross-section plane.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p> <p>Warning: zero or negative scale values not allowed.</p>
orientation	<p>[orientation: accessType initializeOnly, type MFRotation CDATA "0 0 1 0"]</p> <p>orientation is a list of axis-angle orientation 4-tuples applied at each spine-aligned cross-section plane.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p>
beginCap	<p>[beginCap: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Whether beginning cap is drawn (similar to Cylinder top cap).</p> <p>Warning: cannot be changed after initial creation.</p>
endCap	<p>[endCap: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Whether end cap is drawn (similar to Cylinder end cap).</p> <p>Warning: cannot be changed after initial creation.</p>
ccw	<p>[ccw: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>ccw = counterclockwise: ordering of vertex-coordinates orientation.</p> <p>Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.</p>
convex	<p>[convex: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Whether all polygons in a shape are convex (true), or possibly concave (false). A convex polygon is planar, does not intersect itself, and has all interior angles < 180 degrees.</p> <p>Warning: concave geometry may be invisible default convex=true.</p>

creaseAngle	<p>[creaseAngle: accessType initializeOnly, type SFFloat CDATA "0.0"]</p> <p>[0..infinity) creaseAngle defines angle (in radians) where adjacent polygons are drawn with sharp edges or smooth shading. If angle between normals of two adjacent polygons is less than creaseAngle, smooth shading is rendered across the shared line segment.</p> <p>Hint: creaseAngle=0 means render all edges sharply, creaseAngle=3.14 means render all edges smoothly.</p>
solid	<p>[solid: accessType initializeOnly, type SFBool (true false) "true"]</p> <p>Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off).</p> <p>Warning: default value true can completely hide geometry if viewed from wrong side!</p>
set_crossSection	<p>[set_crossSection: accessType inputOnly, type MFVec2f CDATA #FIXED ""]</p> <p>An ordered set of 2D points drawing a piecewise-linear curve and forming a planar series of connected vertices. This provides a silhouette of the outer surface.</p> <p>Warning: match clockwise/counterclockwise or impossible/inverted geometry can result!</p>
set_orientation	<p>[set_orientation: accessType inputOnly, type MFRotation CDATA #FIXED ""]</p> <p>orientation is a list of axis-angle orientation 4-tuples applied at each spine-aligned cross-section plane.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p>
set_scale	<p>[set_scale: accessType inputOnly, type MFVec2f CDATA #FIXED ""]</p> <p>(0..infinity) scale is a list of 2D-scale parameters applied at each spine-aligned cross-section plane.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p> <p>Warning: zero or negative scale values not allowed.</p>
set_spine	<p>[set_spine: accessType inputOnly, type MFVec3f CDATA #FIXED ""]</p> <p>spine is a list of 3D points for a piecewise-linear curve forming a series of connected vertices, open or closed. This is the path along which the crossSection is extruded.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape.</p> <p>containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

Additional Resources

Additional Resources

Graphics Gems book series: many algorithms

- <http://www.graphicsgems.org>

Journal of Graphics Tools (jgt): many algorithms

- <http://jgt.akpeters.com>

Chapter Summary

Chapter Summary 1

Polygonal geometry is essence of X3D graphics

- Also for most other computer graphics approaches

The rendering of almost every geometric shape is usually based on tessellation into triangles

Working with examples is the best way to learn.

Be patient, the principles are consistent, and practice helps make principles familiar.

Chapter Summary 2

Triangles, single-sided polygons, normal vectors

Common fields: *ccw*, *convex*, *creaseAngle*, etc.

Geometry nodes, part 2

- Color and ColorRGBA
- Coordinate and CoordinateDouble
- PointSet
- IndexedLineSet and LineSet
- IndexedFaceSet
- ElevationGrid
- Extrusion

[back to Table of Contents](#)

References

References 1

X3D: Extensible 3D Graphics for Web Authors
by Don Brutzman and Leonard Daly, Morgan
Kaufmann Publishers, April 2007, 468 pages.

- Chapter 6, Geometry Part 2: Points, Lines and Polygons
- <http://x3dGraphics.com>
- <http://x3dgraphics.com/examples/X3dForWebAuthors>

X3D Examples Help

- <http://www.web3d.org/x3d/content/examples/help.html>

References 2

X3D Scene Authoring Hints

- <http://x3dgraphics.com/examples/X3dSceneAuthoringHints.html>

X3D Graphics Specification

- <http://www.web3d.org/x3d/specifications>
- Also available as help pages within X3D-Edit

VRML 2.0 Sourcebook by Nadeau, Ames,
Moreland

References 3

VRML 2.0 Sourcebook by Andrea L. Ames, David R. Nadeau, and John L. Moreland, John Wiley & Sons, 1996.

- <http://www.wiley.com/legacy/compbooks/vrml2sbk/cover/cover.htm>
- <http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook>
- Chapter 13 – Points Lines Faces
- Chapter 14 – Elevation Grid
- Chapter 15 – Extrusion
- Chapter 16 – Color

Contact

Don Brutzman

brutzman@nps.edu

<http://web.nps.navy.mil/~brutzman>

Code USW/Br, Naval Postgraduate School
Monterey California 93943-5000 USA

1.831.656.2149 voice

1.831.656.7599 fax

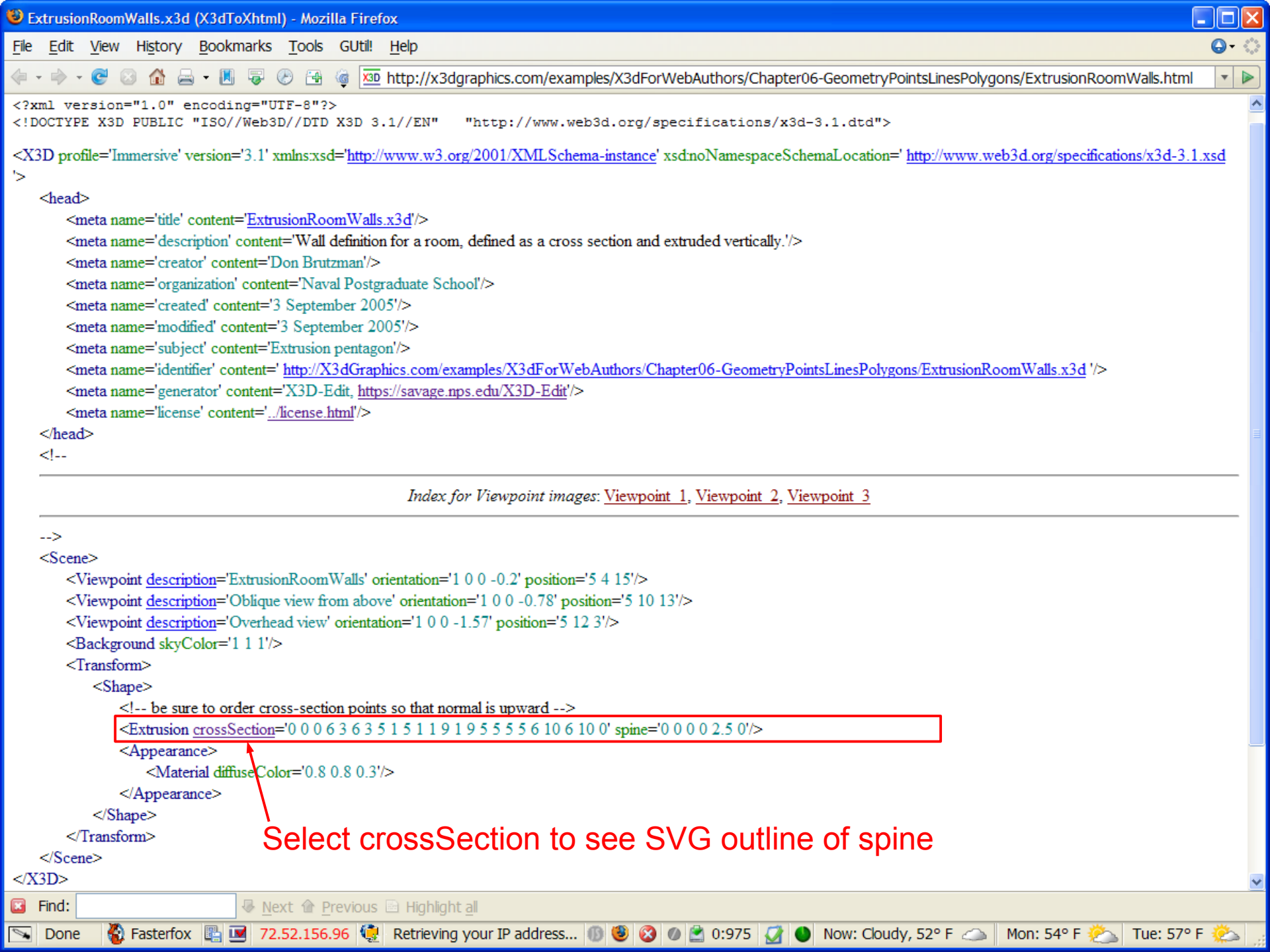
Open-source license

Copyright (c) 1995-2008 held by the author(s). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the names of the Naval Postgraduate School (NPS) Modeling Virtual Environments and Simulation (MOVES) Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.



Extrusion SVG output

X3D Graphics for Web Authors

Chapter 6

Points, Lines and Polygons

Drawing is a struggle between nature and the artist, in which the better the artist understands the intentions of nature, the more easily he will triumph over it. For him it is not a question of copying, but of interpreting in a simpler and more luminous language.

Charles Baudelaire, *On the Ideal and the Model*, 1846.



Contents

Chapter Overview and Concepts

X3D Nodes and Examples

Additional Resources

Chapter Summary

References



Chapter Overview



Many different geometry nodes

An excellent aspect of X3D is that there are many different ways to create geometry

- Chapter 2, Geometry Primitives
- Chapter 6, Points, Lines and Polygons
- Chapter 10, Geometry2D Nodes
- Chapter 13, Triangles and Quadrilaterals



The primitive geometry nodes are *tessellated* (turned into triangles) by X3D browsers.

The primitive geometry nodes might also be written using IndexedFaceSet representations. This would let authors control how many polygons are used, meaning that higher fidelity to curved surfaces and higher-resolution shapes can be produced than might be produced by a given browser.

Fundamental geometry nodes

Geometry nodes in this chapter include
points, lines, and indexed face sets

These nodes fundamental and can represent
almost any shape

- Tools can convert other geometry to simpler forms
- Thus most are part of Interchange profile for
broadest possible usage and adaptability



Tip: Xj3D viewer in X3D-Edit includes wireframe (line) and point rendering modes

- Alt-shift-D on selected Xj3D window to undock it from the X3D-Edit frame
- Alt-shift-W Wireframe rendering
- Alt-shift-P Point rendering

Overview: Points, Lines and Polygons

Triangles, single-sided polygons, normal vectors

Common fields: *ccw*, *convex*, *creaseAngle*, etc.

Geometry nodes, part 2

- Color and ColorRGBA
- Coordinate and CoordinateDouble
- PointSet
- IndexedLineSet and LineSet
- IndexedFaceSet
- ElevationGrid
- Extrusion



[back to Table of Contents](#)

Concepts



Triangles

Triangles are the primary low-level geometry construct used by graphics software, hardware

- More complex shapes are reduced to triangles by the rendering software
- A triangle is always planar, allowing the material appearance to fill it

Sometimes quadrilaterals are used, but problem is that values might be non-coplanar due to roundoff (or authoring) error

- Which means that filling in material is ill defined, and not properly or repeatably renderable

web|3D
CONSORTIUM



Graphics hardware is highly optimized to favor triangles, performing something very simple, many times, extremely quickly.

Single-sided polygons

Graphics engines always prefer simplicity in order to achieve maximum run-time performance

- Top 3 graphics hardware considerations: performance, performance, performance

Single-sided polygons take about half the time to draw than double-sided polygons

- So if authors can arrange geometry so that only one side is ever visible, can go single-sided
- Technical term: *backface culling*
- Efficiency is rationale for many X3D default values
- Example: default setting is *solid*='true' for X3D geometry primitives

Normal vectors

The *normal* vector is perpendicular to the face, pointing away from the centroid of polygon

Direction of normal vector defined by order of points defining the polygon and right-hand rule

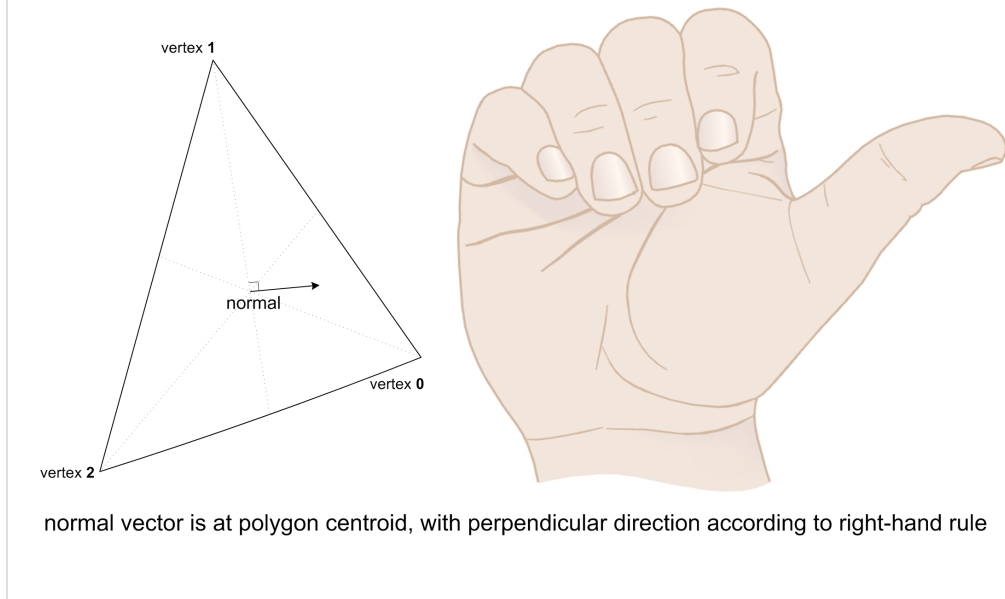
- Align curvature of fingers to match polygon vertex points 0, 1, 2 ...
- Thumb points in direction of (positive) normal, which is the front-facing side
- Negative normal thus points in direction of backface



Normal vectors are only pertinent to polygons, not polyline segments or point sets. The Normal node is covered in Chapter 13, Triangles and Quadrilaterals.

The NormalInterpolator node is covered in Chapter 7, Event Animation.

Right-hand rule for polygon normals



This rule also applies to quadrilaterals and polygons that have more vertices.

Vertices that are not coplanar are degenerate, and can lead to erroneous normal computation and unpredictable rendering results.

Normal vectors are only pertinent to polygons, not polyline segments or point sets. The Normal node is covered in Chapter 13, Triangles and Quadrilaterals.

Common field: *ccw*

ccw (counter clockwise) indicates whether default normal direction of polygons is counterclockwise (default) or clockwise

- *ccw*='true' is right-hand rule
- *ccw*='false' is opposite

Hint: can correct some opposite-rendering geometry by reversing *ccw* value, rather than reordering all coordinates or indices

- Big time saver for some imports

web|3D
CONSORTIUM



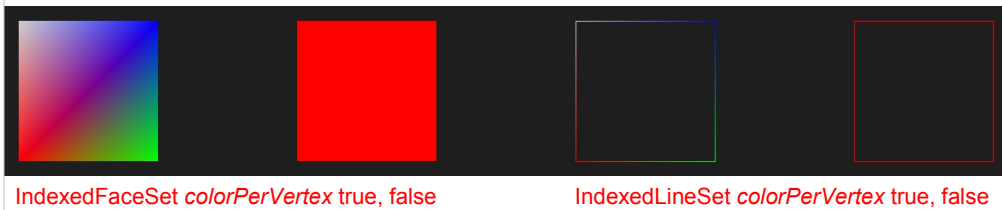
Look at a circular clock face, and apply the right-hand rule at the center with the thumb pointing out from the wall. The curvature of the fingers is counterclockwise.

ccw type is SFBool (boolean).

Common field: *colorPerVertex*

colorPerVertex indicates whether contained color values are applied to each vertex point (default), or to each polygonal face

- *colorPerVertex*='true' requires that # colors must equal # points
- *colorPerVertex*='false' requires that # colors must equal # polygons



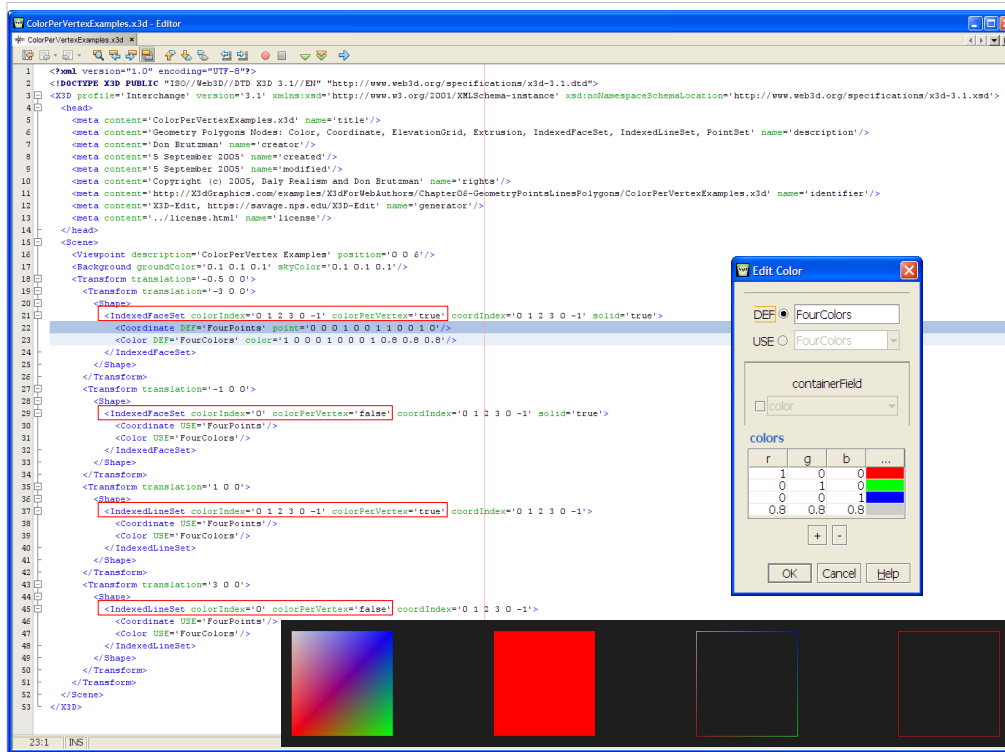
In this context, read # as 'number of'

colorPerVertex type is SFBool (boolean).

Given either value for *colorPerVertex*: counting the number of colors, along with the number of points or polygons, then making sure that those numbers are consistent, is an important correctness check.

X3D for Web Authors, Figure 6.1, p. 160

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ColorPerVertexExamples.x3d>



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ColorPerVertexExamples.x3d>

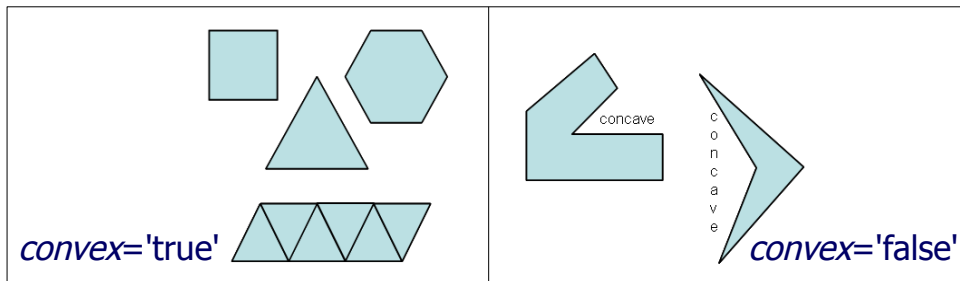
In addition to illustrating *colorPerVertex* effects, this example shows how *Coordinate* and *Color* nodes can be shared by both *IndexedFaceSet* and *IndexedLineSet*.

Also note how each set of *coordIndex* and *colorIndex* fields form a closed loop, both starting and ending with index 0 (prior to the end-of-polygon/end-of-polyline sentinel value -1).

Common field: *convex*

convex indicates whether an n -sided polygon has concave sides, meaning empty-space cavities

- *convex*='true' (default) means no concave sides
- *convex*='false' means concave sides may exist in the polygon, so extra care is needed to avoid hardware or software difficulty when rendering



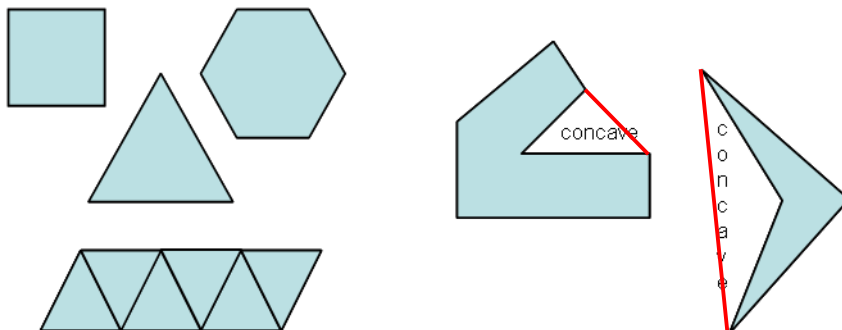
Software or hardware render engines will usually either re-triangulate concave geometry, or else apply more stringent rendering algorithms, in order to avoid mistakenly filling in the cavities found in a non-convex n -sided polygon.

convex type is SFFloat (single-precision floating point).

Note that triangles can only be convex, so triangulation always avoids concave problems.

X3D for Web Authors, Figure 6.2, p. 161

Visual test for concave geometry: whether a line segment drawn between any two of a polygon's vertices intersects space outside of the polygon.



Common field: *creaseAngle*

creaseAngle defines the angle (in radians) used to determine whether adjacent polygons are drawn with sharp edges or smooth shading

- If angle between polygons is less than *creaseAngle*, then smooth shading is used
- Smooth shading can conceal underlying tessellation

creaseAngle only affects shading within a single geometric shape, not exterior boundaries

- *creaseAngle*='0.0' means all edges are sharp
- *creaseAngle*='3.14159' means no edges are sharp



3.14159 = pi = 180 degrees

It is not possible for adjacent polygons cannot have a relative angle > 180 degrees.

creaseAngle type is SFFloat (single-precision floating point).

Browsers perform *creaseAngle* computations by comparing normal vectors.

Many shapes look better with smooth shading. Do not hesitate to experiment with using *creaseAngle*='0' or another small (radian) value to improve visual realism.

Common field: *normalPerVertex*

normalPerVertex indicates whether contained normal values are applied to each vertex point (default), or to each polygonal face

- *normalPerVertex*='true' requires that # normals must equal # points
- *normalPerVertex*='false' requires that # normals must equal # polygons



In this context, read # as 'number of'

normalPerVertex type is SFBool (boolean).

Given either value for *normalPerVertex*: counting the number of normals, along with the number of points or polygons, then making sure that those numbers are consistent, is an important correctness check.

Normal vectors are only pertinent to polygons, not polyline segments or point sets. The Normal node is covered in Chapter 13, Triangles and Quadrilaterals.

Common index fields:

colorIndex, coordIndex, normalIndex

colorIndex, coordIndex, normalIndex

each provide arrays of integer indices that correlate Color, Coordinate or Normal values to individual vertices or polygons

- Initial index is 0
- Sentinel value -1 concludes polygon, polyline
- Maximum value equals (count - 1)
- Type MFInt32, default is empty array



Normal vectors are only pertinent to polygons, not polyline segments or point sets. The Normal node is covered in Chapter 13, Triangles and Quadrilaterals.

index counting checks

- *colorIndex* must equal (point count - 1) when *colorPerVertex*='true', which is default
- *colorIndex* must equal (polygon count - 1) when *colorPerVertex*='false'
- *normalIndex* must equal (point count - 1) when *normalPerVertex*='true', which is default
- *normalIndex* must equal (polygon count - 1) when *normalPerVertex*='false'



Failure to meet these requirements is an error. Results are often unpredictable since the X3D Specification doesn't provide strict requirements for how to handle errors.

[back to Table of Contents](#)

X3D Nodes and Examples



Color node

Color values for individual polygons, line segments and points can be defined using the Color node

Color values are red-green-blue (RGB) [0..1]

- Type is MFColor array of 3-tuple values
- HTML, SVG colors are [0..255] [#000000..#FFFFFF] and so must be converted numerically if used

Appearance and Material node can also be used to control overall transparency, if needed



Commas are allowed as whitespace characters, but the X3D Schema will return an XML validation error if a comma appears within a given SFColor 3-tuple value.

X3D Canonicalization (C14N) reformatting removes intermediate comma characters.

[#000000..#FFFFFF] are hexadecimal (base 16) values.

Color and ColorRGBA nodes can appear as the color field within the following nodes: PointSet, IndexedLineSet, IndexedFaceSet, IndexedTriangleFanSet, IndexedTriangleSet, IndexedTriangleStripSet, TriangleFanSet, TriangleSet, and TriangleStripSet.

ColorRGBA node

ColorRGBA used similarly to Color node, but adds alpha (opacity) to each red-green-blue value

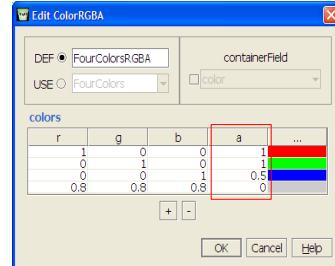
- alpha component equals (1 – transparency)

Alpha values range 0 to 1

- 0 means fully transparent
- 1 means fully opaque

RGBA values selectively allow transparent parts in geometry

- Rather than single Material transparency consistently across full geometry with Color node



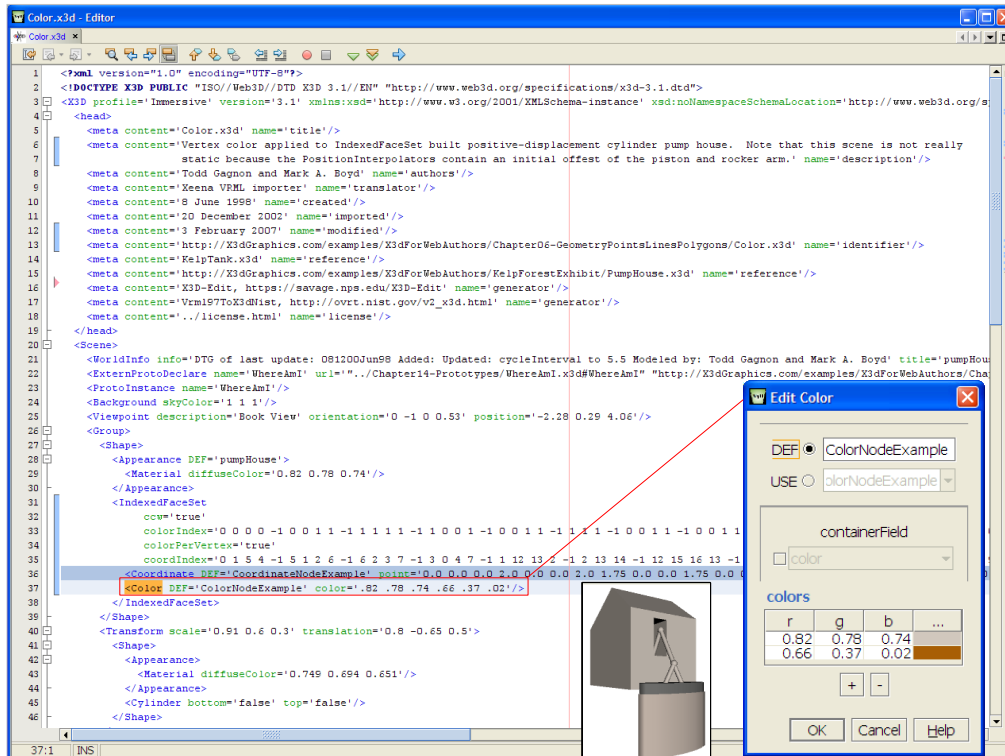
web|3D
CONSORTIUM



Commas are allowed as whitespace characters, but the X3D Schema will return an XML validation error if a comma appears within a given SFCOLORRGBA 4-tuple value.


X3D Canonicalization (C14N) reformatting removes intermediate comma characters.

Color and ColorRGBA nodes can appear as the color field within the following nodes: PointSet, IndexedLineSet, IndexedFaceSet, IndexedTriangleFanSet, IndexedTriangleSet, IndexedTriangleStripSet, TriangleFanSet, TriangleSet, and TriangleStripSet.



X3D for Web Authors, Figure 6.3, p. 164

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/Color.x3d>

Color	<p>Color node defines a set of RGB color values. Color is only used by ElevationGrid, IndexedFaceSet, IndexedLineSet, LineSet and PointSet.</p> <p>Hint: colors are often controlled by Material instead.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
color	<p>[color: accessType inputOutput, type MFColor CDATA #IMPLIED]</p> <p>color defines a set of RGB colors.</p>
containerField	<p>[containerField: NMTOKEN "color"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>
 ColorRGBA	<p>ColorRGBA node defines a set of RGBA color values. ColorRGBA is only used by ElevationGrid, IndexedFaceSet, IndexedLineSet, LineSet and PointSet.</p> <p>Hint: colors are often controlled by Material instead.</p> <p>Hint: alpha channel may be ignored under Interchange profile.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
color	<p>[color: accessType inputOutput, type MFColorRGBA CDATA #IMPLIED]</p> <p>color defines a set of RGBA colors.</p>
containerField	<p>[containerField: NMTOKEN "color"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

<http://www.web3d.org/x3d/content/X3dTooltips.html#Color>

<http://www.web3d.org/x3d/content/X3dTooltips.html#ColorRGBA>

Coordinate node

Provide array of x-y-z point values

- Type MFVec3f array of 3-tuple values, each with 32-bit single-precision floating point

Coordinate *point* values define all of the vertices needed to build polygonal geometry

- *coordIndex* array indicates connectivity for each individual polygon in the shape
- *coordIndex* value -1 indicates end of one polygon, next *coordIndex* value indicates vertex point that begins a new polygon



Commas are allowed as whitespace characters, but the X3D Schema will return an XML validation error if a comma appears within a given SFVec3f 3-tuple value.

X3D Canonicalization (C14N) reformatting removes intermediate comma characters.

Coordinate and CoordinateDouble nodes can appear as the coord field within the following nodes: PointSet, IndexedLineSet, IndexedFaceSet, IndexedTriangleFanSet, IndexedTriangleSet, IndexedTriangleStripSet, TriangleFanSet, TriangleSet, and TriangleStripSet.

CoordinateDouble node

Definition and usage similar to Coordinate node

Provide array of x-y-z point values

- Type MFVec3d array of 3-tuple values, each with 64-bit double-precision floating point

Double precision may be needed for specialty applications (geographic, atomic, etc.)

Note however that most graphics hardware is exclusively single-point precision, for speed

- So browser may need special software techniques to handle double precision fidelity properly



Commas are allowed as whitespace characters, but the X3D Schema will return an XML validation error if a comma appears within a given SFVec3d 3-tuple value.

X3D Canonicalization (C14N) reformatting removes intermediate comma characters.

Coordinate and CoordinateDouble nodes can appear as the coord field within the following nodes: PointSet, IndexedLineSet, IndexedFaceSet, IndexedTriangleFanSet, IndexedTriangleSet, IndexedTriangleStripSet, TriangleFanSet, TriangleSet, and TriangleStripSet.

Coordinate	Coordinate builds geometry using a set of 3D coordinates. Coordinate is used by IndexedFaceSet, IndexedLineSet, LineSet and PointSet. Coordinate is also used by NurbsPositionInterpolator and NurbsOrientationInterpolator.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. <i>Hint:</i> descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. <i>Hint:</i> USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
point	[point: accessType inputOutput, type MFVec3f CDATA #IMPLIED] point contains a set of 3D coordinates.
containerField	[containerField: NMTOKEN "coord"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

CoordinateDouble	CoordinateDouble builds geometry using a set of 3D coordinates. CoordinateDouble is used by IndexedFaceSet, IndexedLineSet, LineSet and PointSet. CoordinateDouble is also used by NurbsPositionInterpolator and NurbsOrientationInterpolator.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. <i>Hint:</i> descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. <i>Hint:</i> USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
point	[point: accessType inputOutput, type MFVec3d CDATA #IMPLIED] point contains a set of 3D coordinates.
containerField	[containerField: NMTOKEN "coord"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#Coordinate>

<http://www.web3d.org/x3d/content/X3dTooltips.html#CoordinateDouble>

PointSet node

PointSet creates a series of simple unconnected points in 3D space

- Contains Coordinate node for *point* data
- Since points are separate, *coordIndex* unnecessary

Each point typically drawn as a single pixel

- Or consistently as multiple pixels
- Thus scaling and perspective are quite deceiving
- **Rarely used** due to perspective inconsistencies

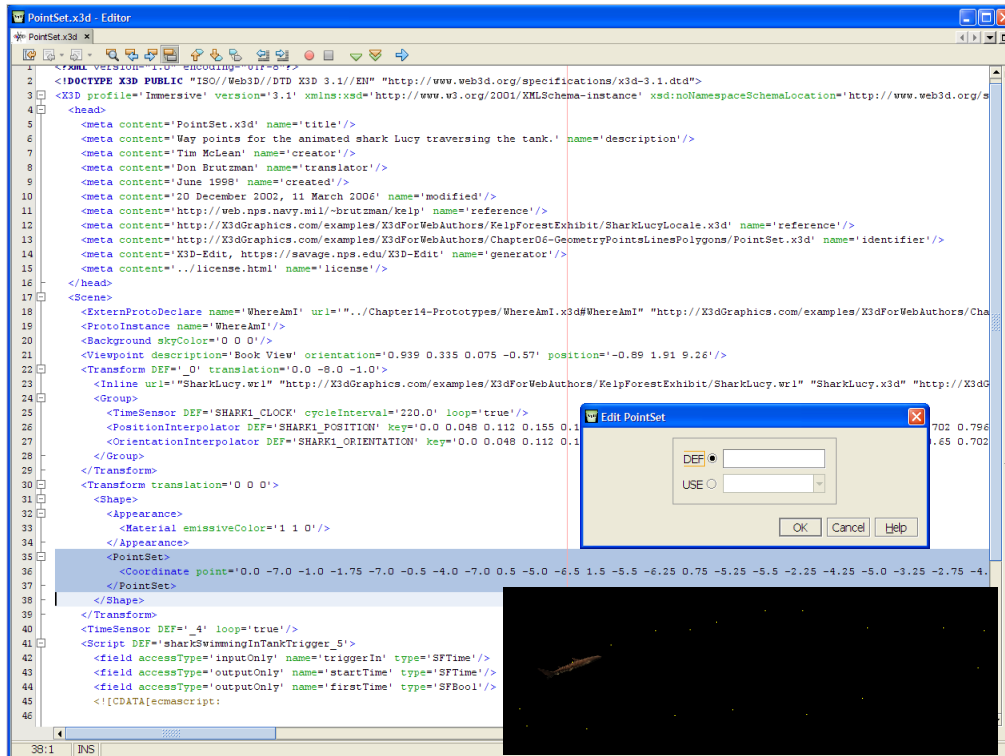
Color can be set in one of two ways

- Uniformly via Material *emissiveColor* value
- Individually via contained Color/ColorRGBA node

Note that Material *diffuseColor* and other fields have no effect on PointSet node.


The term *pixel* is an abbreviation for picture element, that is, the smallest drawable color on a display screen. Because pixel size can vary quite widely depending on screen resolution, authors thus have little control over the actual on-screen size of each point. Different displays and different resolutions can make points look quite different.

Points are not lit, are not texture-mapped, and do not participate in collision detection.



X3D for Web Authors, Figure 6.4, p. 167

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/PointSet.x3d>

 PointSet	<p>PointSet is a node that contains a set of colored 3D points, represented by contained Color and Coordinate nodes. Color values or a Material emissiveColor is used to draw lines and points.</p> <p>Hint: use a different color (or emissiveColor) than the background color.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape.</p> <p>containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

<http://www.web3d.org/x3d/content/X3dTooltips.html#PointSet>

IndexedLineSet node

IndexedLineSet creates an array of line segments

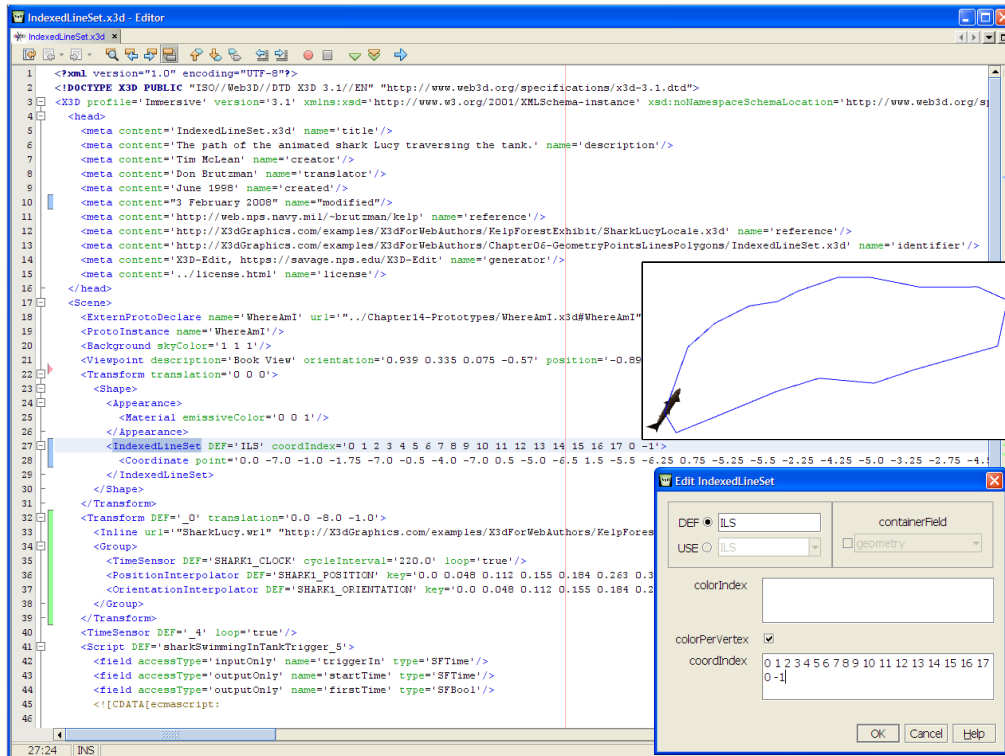
- Contains Coordinate node for *point* data
- Can be discontinuous or share points repeatedly
- Each set of connected line segments is a *polyline*

Lines are not lit, are not texture-mapped, and do not participate in collision detection

Color can be set in one of two ways


- Uniformly via Material *emissiveColor* value
- Individually via contained Color/ColorRGBA node; applied either by individual points, or by each segment, as determined by *colorPerVertex*

Note that Material *diffuseColor* and other fields have no effect on IndexedLineSet.



X3D for Web Authors, Figure 6.5, p. 170

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/IndexedLineSet.x3d>

 IndexedLineSet	<p>IndexedLineSet is a geometry node that can contain a Color node and a Coordinate node. Color values or a Material emissiveColor is used to draw lines and points. Lines are not lit, are not texture-mapped, and do not participate in collision detection.</p> <p>Hint: use a different color (or emissiveColor) than the background color.</p> <p>Hint: if rendering Coordinate points originally defined for an IndexedFaceSet, index values may need to repeat each initial vertex to close each polygon outline. Step-wise colors or linear interpolation of colors can be used as a good scientific visualization technique to map arbitrary function values to a color map.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
coordIndex	<p>[coordIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets, use -1 to separate indices for each polyline.</p> <p>Hint: if rendering Coordinate points originally defined for an IndexedFaceSet, index values may need to repeat initial each initial vertex to close the polygons.</p>
colorPerVertex	<p>[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"]</p> <p>Whether Color node is applied per vertex (true) or per polyline (false).</p>
colorIndex	<p>[colorIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED]</p> <p>colorIndex indices provide order in which colors are applied.</p> <p>Hint: if rendering Coordinate points originally defined for an IndexedFaceSet, index values may need to repeat initial each initial vertex to close the polygons.</p>
set_coordIndex	<p>[set_coordIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""]</p> <p>coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.</p>
set_colorIndex	<p>[set_colorIndex: accessType initializeOnly, type MFInt32 CDATA #FIXED ""]</p> <p>colorIndex indices provide order in which colors are applied.</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedLineSet>

LineSet node

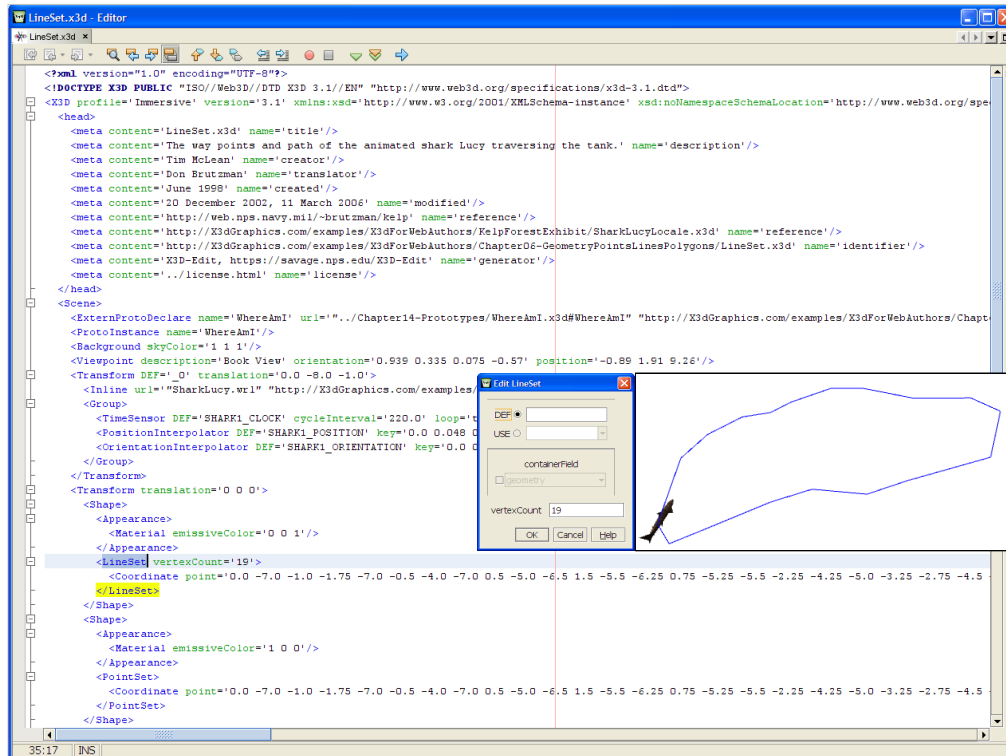
Similar to IndexedLineSet

- Contain 0 or 1 Coordinate/CoordinateDouble
- Material *emissiveColor* or Color/ColorRGBA

Rather than using coordIndex and colorIndex,
LineSet has *vertexCount* field


- *vertexCount* MFInt32 array of integers defines number of sequential points used in each polyline
- No -1 sentinel values needed
- Color and Coordinate values used in defined order
- Somewhat more compact than using indices

Note that Material *diffuseColor* and other fields have no effect on LineSet node.



X3D for Web Authors, Figure 6.5, p. 170

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/IndexedLineSet.x3d>

 LineSet	<p>LineSet is a geometry node that can contain a Color node and a Coordinate node. Color values or a Material emissiveColor is used to draw lines and points. Lines are not lit, are not texture-mapped, and do not participate in collision detection.</p> <p>Hint: use a different color (or emissiveColor) than the background color. Linear interpolation of colors can be used as a good scientific visualization technique to map arbitrary function values to a color map.</p> <p>Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.</p>
DEF	<p>[DEF ID #IMPLIED]</p> <p>DEF defines a unique ID name for this node, referencable by other nodes.</p> <p>Hint: descriptive DEF names improve clarity and help document a model.</p>
USE	<p>[USE IDREF #IMPLIED]</p> <p>USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children.</p> <p>Hint: USEing other geometry (instead of duplicating nodes) can improve performance.</p> <p>Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!</p>
vertexCount	<p>[vertexCount: accessType initializeOnly, type MInt32 CDATA #IMPLIED]</p> <p>(2,infinity) vertexCount describes how many vertices are used in each polyline from Coordinate field. Coordinates are assigned to each line by taking vertexCount[n] vertices from Coordinate field.</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape.</p> <p>containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

<http://www.web3d.org/x3d/content/X3dTooltips.html#LineSet>

IndexedFaceSet node 1

IndexedFaceSet creates a set of polygons (faces)

- Contains Coordinate node for *point* data
- Can be discontinuous or share points repeatedly

Color can be set in one of two ways

- Uniformly via sibling Material fields
- Individually via contained Color/ColorRGBA node; applied either by individual points, or by each polygon, as determined by *colorPerVertex*



All Material fields are active, relevant for IndexedFaceSet and other polygonal nodes.

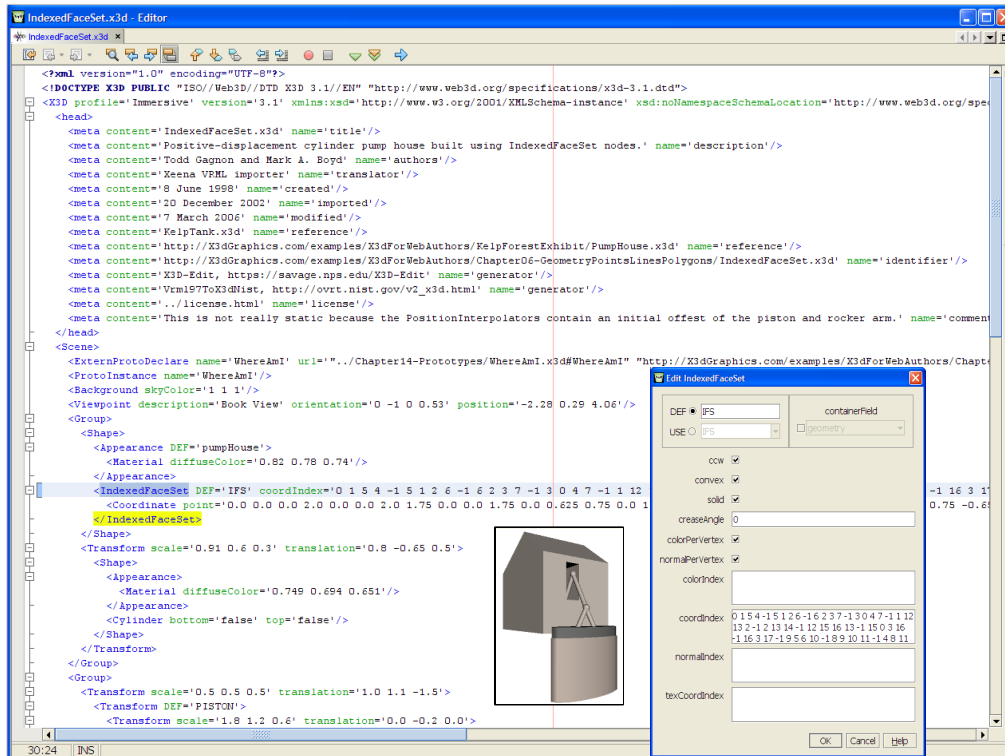
IndexedFaceSet node 2

Many fields and features apply

- *ccw*, *convex*, *solid*, *creaseAngle* as before
- *colorPerVertex*, *normalPerVertex* as before
- *coordIndex*, *colorIndex*, *normalIndex* as before
- *texCoordIndex* applies texture coordinates to map texture images to individual geometry points


Contained nodes (0 or 1 of each)

- Color/ColorRGBA
- Coordinate/CoordinateDouble
- Normal, TextureCoordinate



X3D for Web Authors, Figure 6.7, p. 175

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/IndexedFaceSet.x3d>

 IndexedFaceSet	IndexedFaceSet is a geometry node that can contain a Color, Coordinate, Normal and TextureCoordinate node. Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
coordIndex	[coordIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED] coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.
ccw	[ccw: accessType initializeOnly, type SBool (true/false) "true"] ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
convex	[convex: accessType initializeOnly, type SBool (true/false) "true"] Whether all polygons in a shape are convex (true), or possibly concave (false) A convex polygon is planar, does not intersect itself, and has all interior angles < 180 degrees. Interchange profile hint: only convex=true IndexedFaceSets are supported. Warning: concave geometry may be invisible default convex=true.
solid	[solid: accessType initializeOnly, type SBool (true/false) "true"] Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). Warning: default value true can completely hide geometry if viewed from wrong side!
creaseAngle	[creaseAngle: accessType initializeOnly, type SFloat CDATA "0"] [0..infinity) creaseAngle defines angle (in radians) for determining whether adjacent polygons are drawn with sharp edges or smooth shading. If angle between normals of two adjacent polygons is less than creaseAngle, smooth shading is rendered across the shared line segment. Interchange profile hint: only 0 and π radians supported. Hint: creaseAngle=0 means render all edges sharply, creaseAngle=3.14 means render all edges smoothly.
colorPerVertex	[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"] Whether Color node is applied per vertex (true) or per polygon (false).
colorIndex	[colorIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED] colorIndex indices provide order in which colors are applied.
normalPerVertex	[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"] Whether Normal node is applied per vertex (true) or per polygon (false).
normalIndex	[normalIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED] Interchange profile hint: this field may be ignored.

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedFaceSet>

texCoordIndex	[texCoordIndex: accessType initializeOnly, type MFInt32 CDATA #IMPLIED] List of texture-coordinate indices mapping attached texture to corresponding coordinates. Hint : use a tool!
set_coordIndex	[set_coordIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""] coordIndex indices provide order in which coordinates are applied. Order starts at index 0, commas are optional between sets. Use -1 to separate indices for each polygon.
set_colorIndex	[set_colorIndex: accessType initializeOnly, type MFInt32 CDATA #FIXED ""] colorIndex indices provide order in which colors are applied.
set_normalIndex	[set_normalIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""] Interchange profile hint : this field may be ignored.
set_texCoordIndex	[set_texCoordIndex: accessType inputOnly, type MFInt32 CDATA #FIXED ""] List of texture-coordinate indices mapping attached texture to corresponding coordinates. Hint : use a tool!
containerField	[containerField: NMTOKEN "geometry"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#IndexedFaceSet>

ElevationGrid node

ElevationGrid takes a rectangular array of floats and converts *height* array into post values above (or below) baseline $y=0$ ground plane

- *xSpacing, zSpacing* are lengths in meters
- *xDimension, zDimension* are row, column sizes
- *height* MFFloat array size $xDimension \cdot zDimension$
- *ccw, solid* as before
- *colorPerVertex, normalPerVertex* as before

Contained nodes (0 or 1 of each)

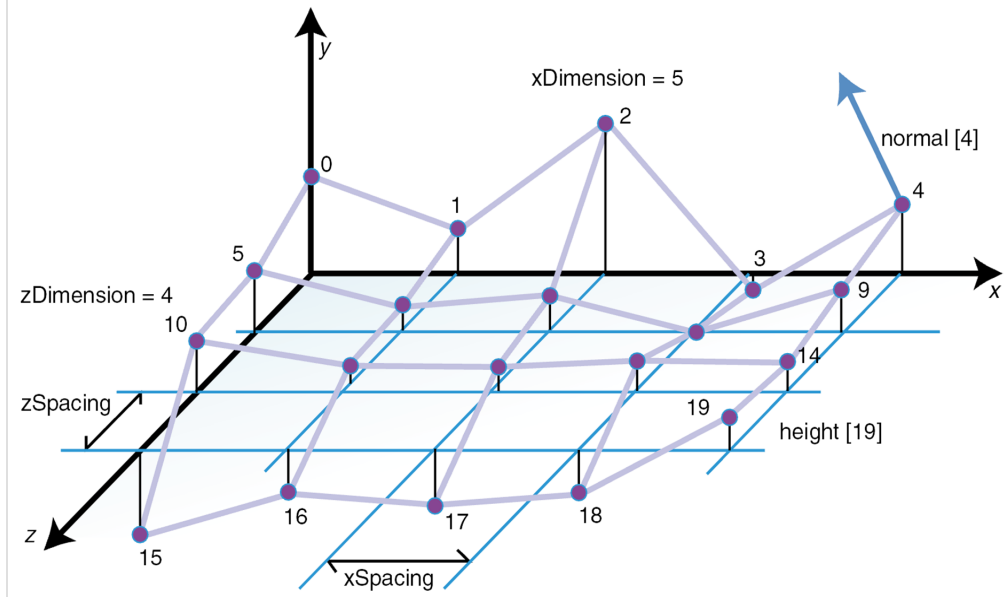
- Color/ColorRGBA, Normal, TextureCoordinate



ElevationGrid is a good way to create terrain and some special shapes.

Related node: GeoElevationGrid in X3D Geospatial Component.

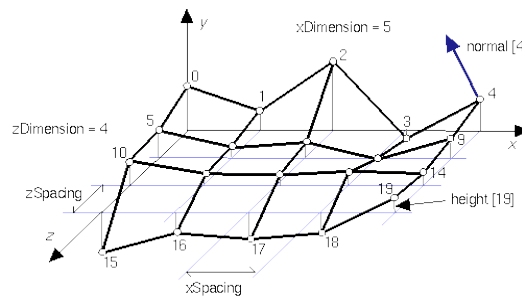
ElevationGrid indexing of *height* array



Indexing scheme for the ElevationGrid *height* array, including relationship to *xDimension/zDimension* and *xSpacing/zSpacing*.

X3D for Web Authors, Figure 6.8, p. 177

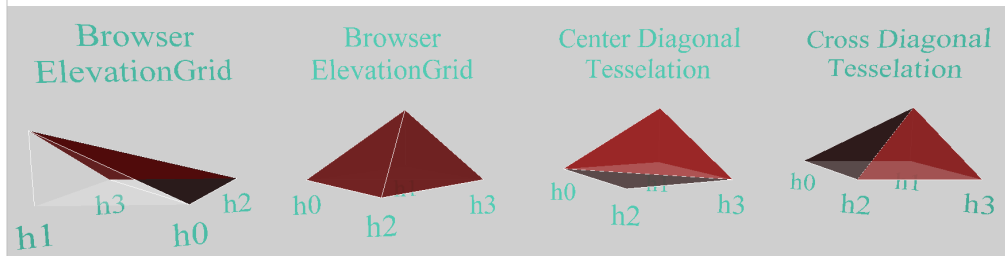
Adapted with permission from X3D Abstract Specification, section 13.3.4 ElevationGrid, Figure 13.4.



ElevationGrid node

Alternate forms of tessellation are possible for nonplanar ElevationGrid quadrilaterals

- Almost all ElevationGrid quads are nonplanar, otherwise the geometry is flat
- Leftmost two figures show different views of grid
- Rightmost two figures show different tessellations
- Usually not a problem for large, fine-scaled grid




X3D for Web Authors, Figure 6.9, p. 178

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ElevationGridNonPlanarQuadrilaterals.x3d>

ElevationGrid node X3D-Edit

[example TBD]

 ElevationGrid	ElevationGrid is a geometry node. ElevationGrid is a rectangular grid of varying height above a flat surface. ElevationGrid can contain Color, Normal and TextureCoordinate nodes. Hint: insert a Shape node before adding geometry or Appearance. You can also substitute a type-matched ProtoInstance for content.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring all other attributes and children. Hint: USING other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
xDimension	[xDimension: accessType initializeOnly, type SInt32 CDATA "0"] Number of grid-array elements along X direction.
zDimension	[zDimension: accessType initializeOnly, type SInt32 CDATA "0"] Number of grid-array elements along Z direction.
xSpacing	[xSpacing: accessType initializeOnly, type SFloat CDATA "1.0"] Meters distance between grid-array vertices along X direction. Hint: total horizontal x-axis distance equals (xDimension-1) * xSpacing.
zSpacing	[zSpacing: accessType initializeOnly, type SFloat CDATA "1.0"] Meters distance between grid-array vertices along Z direction. Hint: total vertical z-axis distance equals (zDimension-1) * zSpacing.
height	[height: accessType initializeOnly, type MFFloat CDATA #IMPLIED] Grid array of height vertices along upward Y direction, with xDimension rows and zDimension columns.
set_height	[set_height: accessType inputOnly, type MFFloat CDATA #FIXED ""] Grid array of height vertices along upward Y direction, with xDimension rows and zDimension columns.
ccw	[ccw: accessType initializeOnly, type SBool (true/false) "true"] ccw = counterclockwise: ordering of vertex coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
creaseAngle	[creaseAngle: accessType initializeOnly, type SFloat CDATA "0"] (0..infinity) creaseAngle defines angle (in radians) for determining whether adjacent polygons are drawn with sharp edges or smooth shading. If angle between normals of two adjacent polygons is less than creaseAngle, smooth shading is rendered across the shared line segment. Hint: creaseAngle=0 means render all edges sharply, creaseAngle=3.14 means render all edges smoothly.
solid	[solid: accessType initializeOnly, type SBool (true/false) "true"] Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off). Warning: default value true can completely hide geometry if viewed from wrong side!
colorPerVertex	[colorPerVertex: accessType initializeOnly, type SBool (true/false) "true"] Whether Color node is applied per vertex (true) or per quadrilateral (false).
normalPerVertex	[normalPerVertex: accessType initializeOnly, type SBool (true/false) "true"] Whether Normal node is applied per vertex (true) or per quadrilateral (false).
containerField	[containerField: NMTOKEN "geometry"] containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape. containerField attribute is only supported in XML encoding of X3D scenes.
class	[class CDATA #IMPLIED] class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.

<http://www.web3d.org/x3d/content/X3dTooltips.html#ElevationGrid>

Extrusion node 1

Extrusion begins with a planar *crossSection* outline, then stretches (extrudes) it along a *spine* polyline

- *crossSection* is MFVec2f array of 2-tuple floating-point 2D coordinate pairs creating an outline
- *spine* is MFVec3f array of 3-tuple floating-point 3D coordinates creating a polyline

Extrusion is a bit tricky to master, but provides a great way to create sophisticated geometry with little effort

web|3D
CONSORTIUM



Jeff Weekley has a Play-Doh Fun Factory machine to illustrate Extrusion creation.

Play-Doh Fun Factory

- Squeeze, shape, mold and extrude all kinds of crazy shapes with this classic set!
- 2 shapemaking strips have 10 patterns for extruding long strips of PLAY-DOH compound!
- Includes two 5-oz. cans of modeling compound, extruder with 3 half molds, 2 shapemaking strips (10 designs) and trimmer knife.

http://www.hasbro.com/playdoh/default.cfm?page=products&product_id=8994



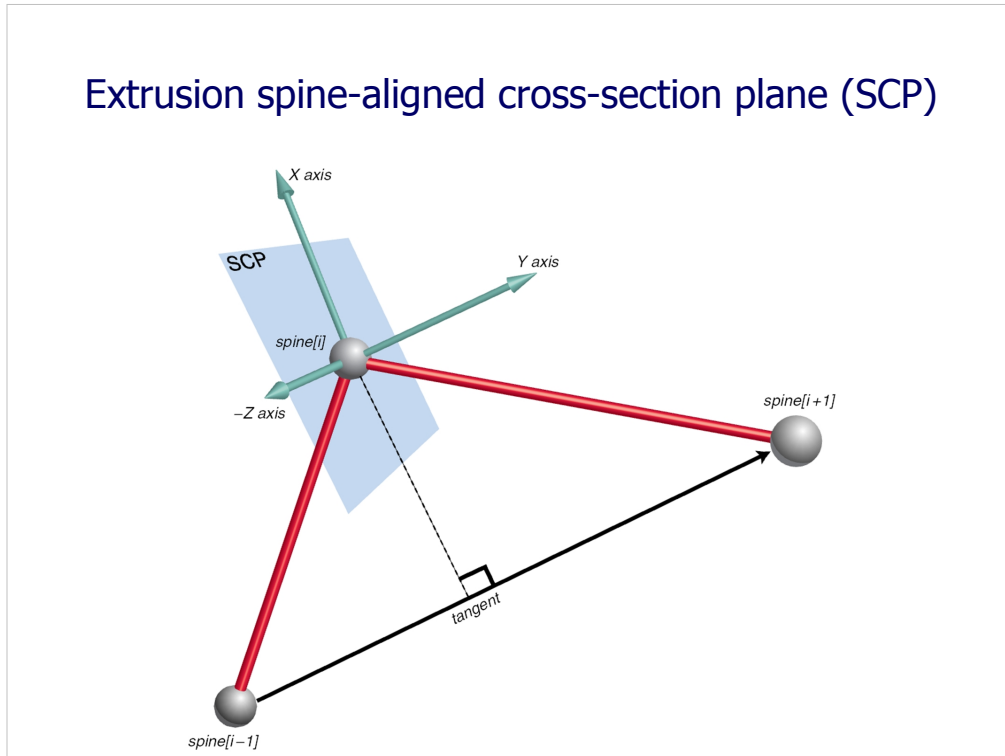
Extrusion node 2

Spine-aligned cross-section plane (SCP) refers to each individual copy of the cross section, each of which appear about each *spine* point

- Extrusion outer hull simply connects corresponding points on these cross-section outlines
- If the outline of the Extrusion is degenerate or ill defined, then the polygons making up Extrusion outline are similarly confounded

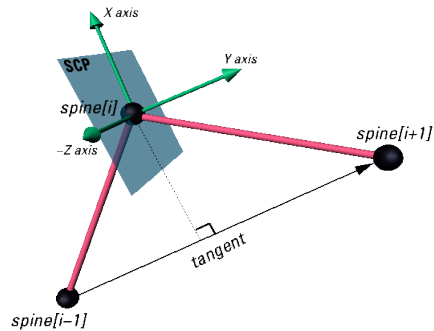
Drawing simple outlines of *crossSection* on graph paper is great way to keep things straight

Extrusion spine-aligned cross-section plane (SCP)



X3D for Web Authors, Figure 6.10, p. 181

Adapted with permission from X3D Abstract Specification, section 13.3.5 Extrusion, Figure 13.5, Spine-aligned cross-section plane at a spine point.



Extrusion node 2

Further modifications include *scale*, *orientation* to modify each cross-section about SCP center

- *scale* is MFVec2f array of 2-tuple floating-point pairs to scale the local spine-aligned crossSection plane
- *orientation* is MFRotation array to rotate
- Single value affects all simultaneously, array affects each repeated cross-section individually

Other fields are common

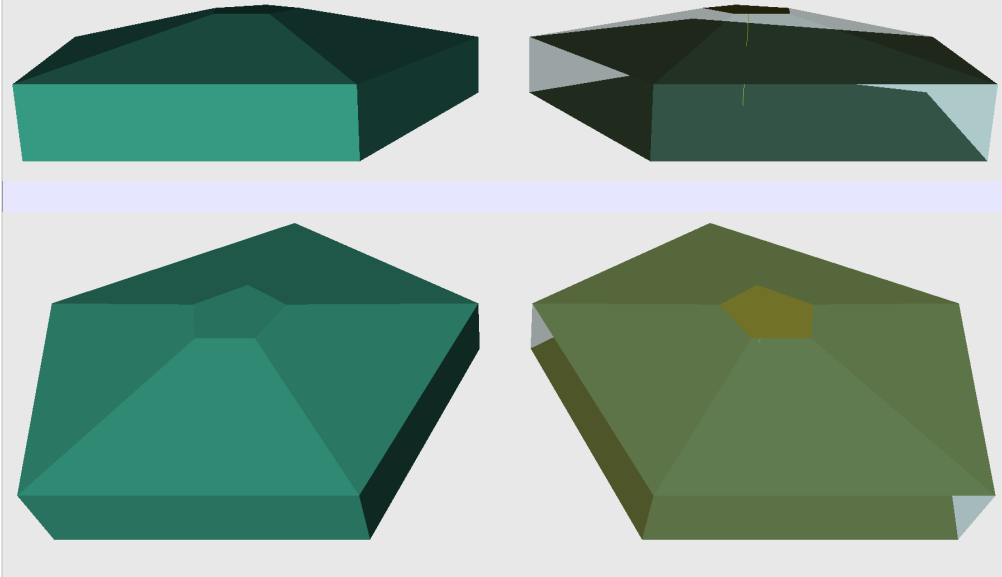
- *ccw*, *convex*, *solid*, *creaseAngle* as before
- *beginCap*, *endCap* are SFBool values to close ends



scale is easy to modify and get good shape changes at each SCP outline. Since it is a 2-tuple MFVec2f array, each scaling pair can be non-uniform.

orientation is quite tricky, only attempt small rotation changes or significant errors can occur (such as geometry folding back on top of itself). Rotations are not cumulative but rather are applied at each of the SCP silhouettes, as indicated in the figure.

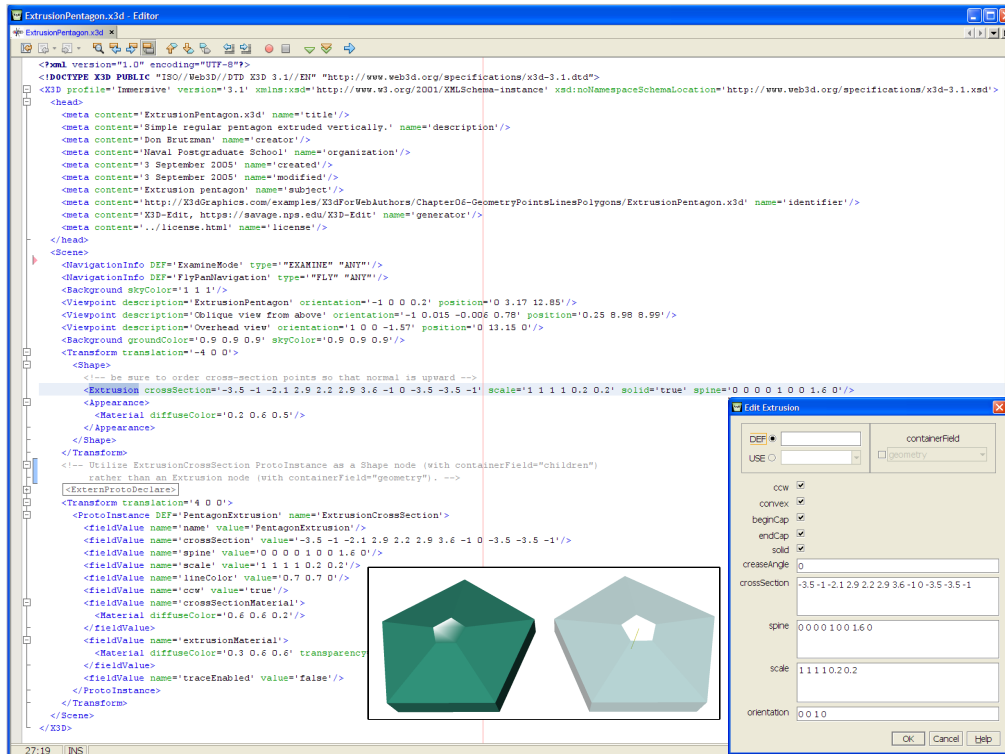
Extrusion cross-section example: pentagon



Example pentagon Extrusion views showing default rendering on left, and cross sections on right, with yellow *spine* line segment visible in upper right.

X3D for Web Authors, Figure 6.11, p. 182

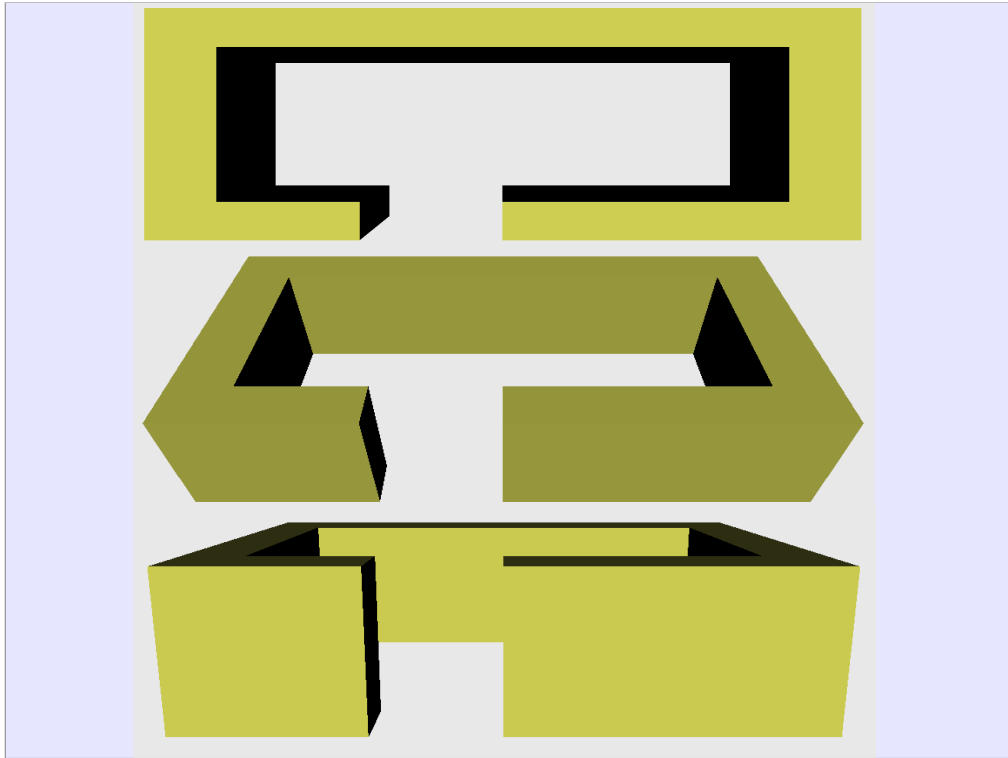
<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionPentagon.x3d>



Example pentagon Extrusion views showing default rendering on left, and cross sections on right.

X3D for Web Authors, Figure 6.11, p. 182

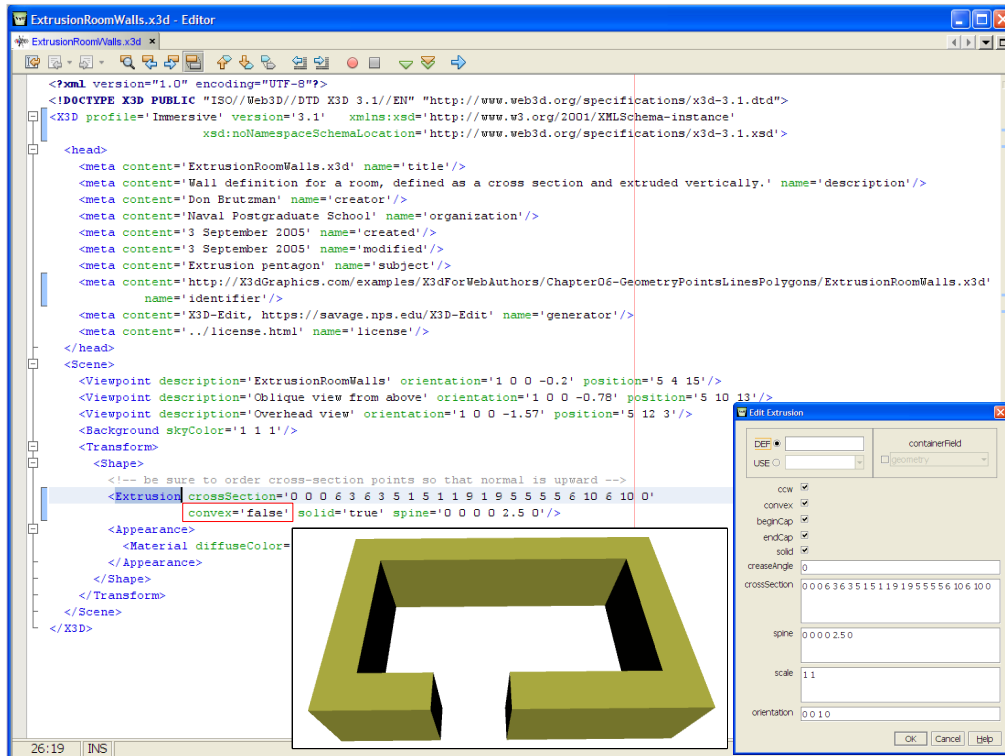
<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionPentagon.x3d>



Extrusion example constructing the walls of a building.

X3D for Web Authors, Figure 6.12, p. 183

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionRoomWalls.x3d>



Extrusion example constructing the walls of a building.

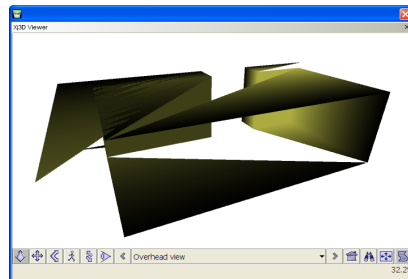
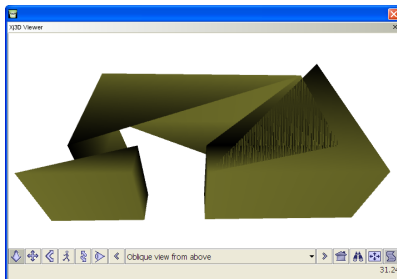
X3D for Web Authors, Figure 6.12, p. 183

<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionRoomWalls.x3d>

Concave geometry effects

Concave geometry with *convex*='true'
can lead to confused geometry results

- Nonsensical polygons
- Aliasing (tearing) of coplanar polygons
- To correct: set *convex*='false'



<http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionRoomWalls.x3d>

Debugging Extrusion problems 1

Unlike most other nodes, ill-defined geometry is possible with Extrusion. Things to check:

- Verify proper *crossSection*, *spine*, *scale*, *orientation* array values and lengths
- Set *convex*='false' if geometry might be concave
- Set *solid*='false' to render inside and outside, eliminating 'invisible geometry' when viewed from behind or inside the exterior hull
- Set *ccw*='false' if crossSection might be defined in clockwise direction

Debugging Extrusion problems 2

Counting checks

- Length of 2-tuple *scale* array must be 0, 1, or match length of 3-tuple *spine* array
- Length of 4-tuple *orientation* array must be 0, 1, or match length of 3-tuple *spine* array
- Values in *scale* and *crossSection* arrays must be multiple of 2 (MFVec2f)
- Values in *spine* array must be multiple of 3 (MFVec3f)
- Values in *orientation* array must be multiple of 4 (MFRotation)

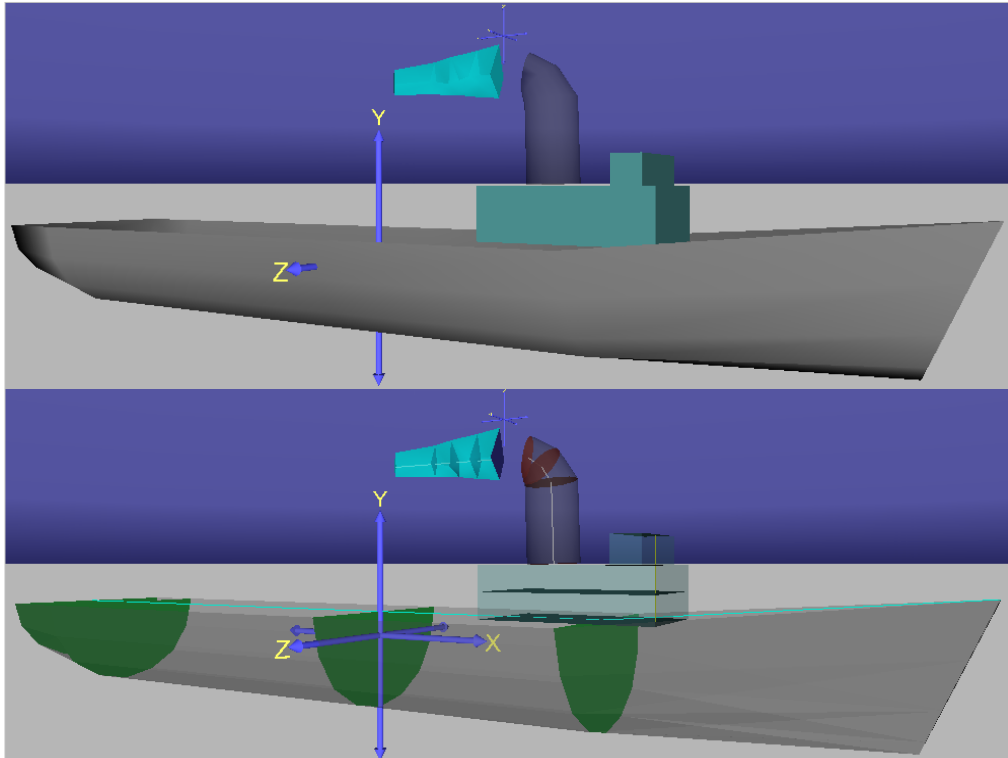
ExtrusionCrossSection prototype

The preceding ExtrusionPentagon.x3d example scene contains new construct: Prototype node

- ExternProtoDeclare refers to external prototype url and defines field signatures
- ProtoInstance creates an instance of new node
- fieldValue definitions provide parameter values, in this case the same values as Extrusion of interest

Result is a specially computed Extrusion showing *crossSection* planes, *spine*, transparent sides

- Can provide helpful insight and debugging support



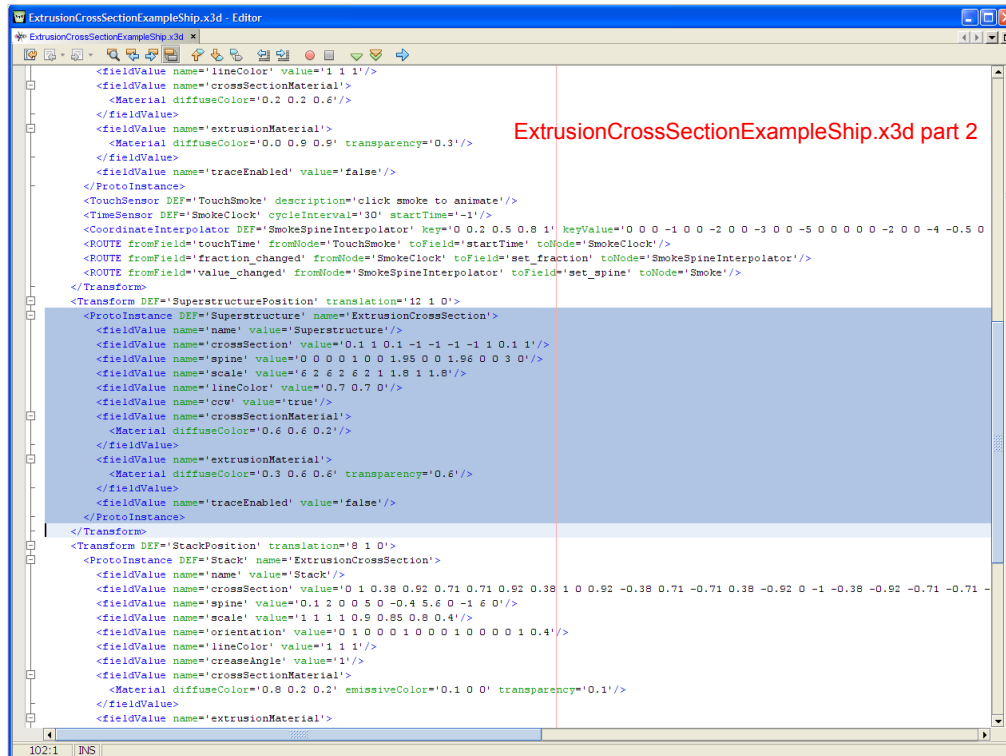
Extrusion example constructing the hull, superstructure, and smoke trail of a ship.

X3D for Web Authors, Figure 6.13, p. 184

<http://www.web3d.org/x3d/content/examples/Basic/course/ExtrusionCrossSectionExampleShip.x3d>

<http://www.web3d.org/x3d/content/examples/Basic/course/ExtrusionCrossSectionPrototype.x3d>

Chapter 14, Creating Prototype Nodes covers ProtoInstance and ExternProtoDeclare.




Extrusion example constructing the hull, superstructure, and smoke trail of a ship.

X3D for Web Authors, Figure 6.13, p. 184

<http://www.web3d.org/x3d/content/examples/Basic/course/ExtrusionCrossSectionExampleShip.x3d>

<http://www.web3d.org/x3d/content/examples/Basic/course/ExtrusionCrossSectionPrototype.x3d>

Chapter 14, Creating Prototype Nodes covers ProtoInstance and ExternProtoDeclare.

 Extrusion	Extrusion is a geometry node stretching a 2D cross section along a 3D-spine path in the local coordinate system Scaling/rotating cross-sections can produce a variety of shapes. Hint: insert a Shape node before adding geometry or Appearance.
DEF	[DEF ID #IMPLIED] DEF defines a unique ID name for this node, referencable by other nodes. Hint: descriptive DEF names improve clarity and help document a model.
USE	[USE IDREF #IMPLIED] USE means reuse an already DEF-ed node ID, ignoring _all_ other attributes and children. Hint: USEing other geometry (instead of duplicating nodes) can improve performance. Warning: do NOT include DEF (or any other attribute values) when using a USE attribute!
spine	[spine: accessType initializeOnly, type MFVec3f CDATA "0 0 0, 0 1 0"] spine is a list of 3D points for a piecewise-linear curve forming a series of connected vertices, open or closed. This is the path along which the crossSection is extruded. Hint: number of spine points, scale values and orientation values must be the same.
crossSection	[crossSection: accessType initializeOnly, type MFVec2f CDATA "1 1, 1 -1, -1 -1, -1 1, 1 1"] An ordered set of 2D points drawing a piecewise-linear curve and forming a planar series of connected vertices. This provides a silhouette of the outer surface. Warning: match clockwise/counterclockwise or impossible/inverted geometry can result!
scale	[scale: accessType initializeOnly, type MFVec2f CDATA "1 1"] (0..infinity) scale is a list of 2D-scale parameters applied at each spine-aligned cross-section plane. Hint: number of spine points, scale values and orientation values must be the same. Warning: zero or negative scale values not allowed.
orientation	[orientation: accessType initializeOnly, type MFRotation CDATA "0 0 1 0"] orientation is a list of axis-angle orientation 4-tuples applied at each spine-aligned cross-section plane. Hint: number of spine points, scale values and orientation values must be the same.
beginCap	[beginCap: accessType initializeOnly, type SFBool (true/false) "true"] Whether beginning cap is drawn (similar to Cylinder top cap). Warning: cannot be changed after initial creation.
endCap	[endCap: accessType initializeOnly, type SFBool (true/false) "true"] Whether end cap is drawn (similar to Cylinder end cap). Warning: cannot be changed after initial creation.
ccw	[ccw: accessType initializeOnly, type SFBool (true/false) "true"] ccw = counterclockwise: ordering of vertex-coordinates orientation. Hint: ccw false can reverse solid (backface culling) and normal-vector orientation.
convex	[convex: accessType initializeOnly, type SFBool (true/false) "true"] Whether all polygons in a shape are convex (true), or possibly concave (false). A convex polygon is planar, does not intersect itself, and has all interior angles < 180 degrees. Warning: concave geometry may be invisible default convex=true.

<http://www.web3d.org/x3d/content/X3dTooltips.html#Extrusion>

creaseAngle	<p>[creaseAngle: accessType initializeOnly, type SFFloat CDATA "0.0"]</p> <p>(0..infinity) creaseAngle defines angle (in radians) where adjacent polygons are drawn with sharp edges or smooth shading. If angle between normals of two adjacent polygons is less than creaseAngle, smooth shading is rendered across the shared line segment.</p> <p>Hint: creaseAngle=0 means render all edges sharply, creaseAngle=3.14 means render all edges smoothly.</p>
solid	<p>[solid: accessType initializeOnly, type SBool (true/false) "true"]</p> <p>Setting solid true means draw only one side of polygons (backface culling on), setting solid false means draw both sides of polygons (backface culling off).</p> <p>Warning: default value true can completely hide geometry if viewed from wrong side!</p>
set_crossSection	<p>[set_crossSection: accessType inputOnly, type MFVec2f CDATA #FIXED ""]</p> <p>An ordered set of 2D points drawing a piecewise-linear curve and forming a planar series of connected vertices. This provides a silhouette of the outer surface.</p> <p>Warning: match clockwise/counterclockwise or impossible/inverted geometry can result!</p>
set_orientation	<p>[set_orientation: accessType inputOnly, type MFRotation CDATA #FIXED ""]</p> <p>orientation is a list of axis-angle orientation 4-tuples applied at each spine-aligned cross-section plane.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p>
set_scale	<p>[set_scale: accessType inputOnly, type MFVec2f CDATA #FIXED ""]</p> <p>(0..infinity) scale is a list of 2D-scale parameters applied at each spine-aligned cross-section plane.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p> <p>Warning: zero or negative scale values not allowed.</p>
set_spine	<p>[set_spine: accessType inputOnly, type MFVec3f CDATA #FIXED ""]</p> <p>spine is a list of 3D points for a piecewise-linear curve forming a series of connected vertices, open or closed. This is the path along which the crossSection is extruded.</p> <p>Hint: number of spine points, scale values and orientation values must be the same.</p>
containerField	<p>[containerField: NMTOKEN "geometry"]</p> <p>containerField is the field-label prefix indicating relationship to parent node. Examples: geometry Box, children Group, proxy Shape.</p> <p>containerField attribute is only supported in XML encoding of X3D scenes.</p>
class	<p>[class CDATA #IMPLIED]</p> <p>class is a space-separated list of classes, reserved for use by XML stylesheets. class attribute is only supported in XML encoding of X3D scenes.</p>

<http://www.web3d.org/x3d/content/X3dTooltips.html#Extrusion>

[back to Table of Contents](#)

Additional Resources



Additional Resources

Graphics Gems book series: many algorithms

- <http://www.graphicsgems.org>

Journal of Graphics Tools (jgt): many algorithms

- <http://jgt.akpeters.com>



[back to Table of Contents](#)

Chapter Summary



Chapter Summary 1

Polygonal geometry is essence of X3D graphics

- Also for most other computer graphics approaches

The rendering of almost every geometric shape
is usually based on tessellation into triangles

Working with examples is the best way to learn.

Be patient, the principles are consistent, and
practice helps make principles familiar.



Chapter Summary 2

Triangles, single-sided polygons, normal vectors

Common fields: *ccw*, *convex*, *creaseAngle*, etc.

Geometry nodes, part 2

- Color and ColorRGBA
- Coordinate and CoordinateDouble
- PointSet
- IndexedLineSet and LineSet
- IndexedFaceSet
- ElevationGrid
- Extrusion



[back to Table of Contents](#)

References



References 1

X3D: Extensible 3D Graphics for Web Authors

by Don Brutzman and Leonard Daly, Morgan Kaufmann Publishers, April 2007, 468 pages.

- Chapter 6, Geometry Part 2: Points, Lines and Polygons
- <http://x3dGraphics.com>
- <http://x3dgraphics.com/examples/X3dForWebAuthors>

X3D Examples Help

- <http://www.web3d.org/x3d/content/examples/help.html>



References 2

X3D Scene Authoring Hints

- <http://x3dgraphics.com/examples/X3dSceneAuthoringHints.html>

X3D Graphics Specification

- <http://www.web3d.org/x3d/specifications>
- Also available as help pages within X3D-Edit

VRML 2.0 Sourcebook by Nadeau, Ames,
Moreland



References 3

VRML 2.0 Sourcebook by Andrea L. Ames,
David R. Nadeau, and John L. Moreland,
John Wiley & Sons, 1996.

- <http://www.wiley.com/legacy/compbooks/vrml2sbk/cover/cover.htm>
- <http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook>
- Chapter 13 – Points Lines Faces
- Chapter 14 – Elevation Grid
- Chapter 15 – Extrusion
- Chapter 16 – Color



Contact

Don Brutzman

brutzman@nps.edu

<http://web.nps.navy.mil/~brutzman>

Code USW/Br, Naval Postgraduate School
Monterey California 93943-5000 USA

1.831.656.2149 voice

1.831.656.7599 fax



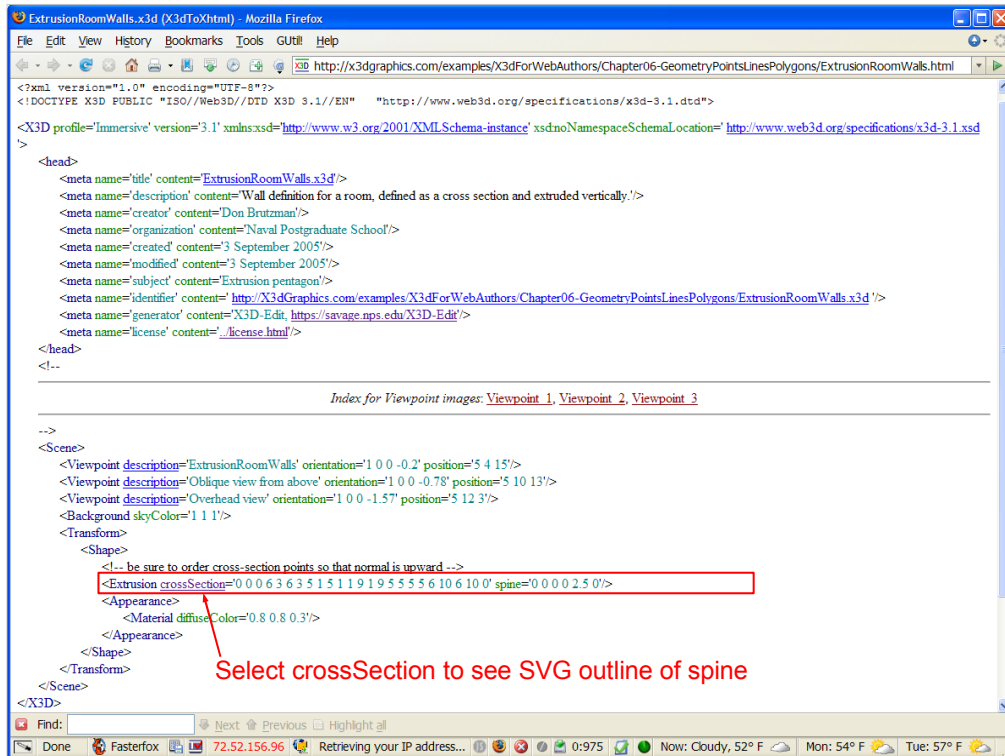
Open-source license

Copyright (c) 1995-2008 held by the author(s). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the names of the Naval Postgraduate School (NPS) Modeling Virtual Environments and Simulation (MOVES) Institute nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter06-GeometryPointsLinesPolygons/ExtrusionRoomWalls.x3d

Extrusion SVG output



```
<?xml version="1.0" encoding="UTF-8"?>
<svg id="ExtrusionRoomWalls.Extrusion1.svg" width="900" height="620">
  <rect width="900" height="620" fill="cyan" stroke="black"/>
  <!-- <x3d:Extrusion crossSection=' 0 0 0 6 3 6 3 5 1 5 1 1 9 1 9 5 5 5 5 6 10 6 10 0'> -->
  <!-- Min xPos = 0 Max xPos = 0 Min yPos = 0 Max yPos = 0 -->
  <g transform="translate(450 310) scale(100 -100)">
    <circle id="marker1" style="fill:red;stroke:red;stroke-width:0.001" cx="0" cy="0 0 6 3 6 3 5 1 5 1 1 9 1 9 5 5 5 5
      6 10 6 10 0" r="0.05"/>
    <set xlink:href="#markerText1" attributeType="CSS" attributeName="visibility" to="visible"
      begin="marker1.mouseover" end="marker1.mouseout"/>
  </g>
  <g id="markerTexts" transform="translate(450 310)">
    <text id="markerText1" style="visibility:hidden;font-size:14;font-weight:bold; fill:green; stroke:none" x="5"
      y="NaN">
      (0,0 0 6 3 6 3 5 1 5 1 1 9 1 9 5 5 5 5 6 10 6 10 0)
    </text>
  </g>
  <g transform="translate(450 310) scale(100 -100)">
    <line x1="0" y1="0" x2="0" y2="0" style="fill:none;stroke:rgb(0,0,0);stroke-width:0.03"/>
    <line x1="0" y1="0" x2="0" y2="0" style="fill:none;stroke:rgb(0,0,0);stroke-width:0.03"/>
  </g>
</svg>
```