X3D Graphics for Web Authors
Chapter 12

Environment Sensor and Sound Nodes

Hereafter, when they come to model heav'n
And calculate the stars, how they will wield
The mighty frame, how build, unbuild, contrive
To save appearances, how gird the sphere
with centric and eccentric scribbled o'er,
Cycle and epicycle, orb in orb.

John Milton, Paradise Lost, 1667

web3D CONSORTIUM

Contents

Chapter Overview

Overview: Environment Sensor and Sound Nodes

Common fields

- *center*, *size*, *enabled*, *isActive*, *enterTime*, *exitTime*

Nodes

- LoadSensor detects availability of other content
- ProximitySensor detects user location, orientation
- VisibilitySensor detects visibility of region to user
- Sound controls spatialization of audio outputs
- AudioClip controls retrieval and playback of audio files and streams

Concepts

review

Common field: *enabled*

enabled is an inputOutput boolean field that turns a sensor node on or off

- Thus allowing author to permit or disable flow of user-driven events which drive other responses
- Set *enabled*='false' to disrupt an event chain

Regardless of whether *enabled*='true' a sensor still needs a ROUTE connection from its output, or else no interaction response occurs

Common field: *isActive*

isActive is an outputOnly boolean field that reports when sensor has received user input

- *isActive* true value sent when proximity or visibility conditions are met
- *isActive* false value sent when proximity or visibility conditions are no longer met

Routing *isActive* values can enable, disable TimeSensor and other animation nodes

- Rapid sequencing on/off can be a difficulty, however
- BooleanFilter, BooleanToggle, BooleanTrigger help, described in Chapter 9 Event Utilities and Scripting

Common fields: *center, size*

center and *size* are SFVec3f fields indicating the location and extent of the box that corresponds to the sensed volume of the node

These values are relative to transformation hierarchy created by parent Transform nodes

- *center* can be scaled, translated, rotated
- *size* can be scaled, rotated

This is helpful for DEF/USE in multiple locations

- Warning: no box overlap, or results unpredictable
- *size*='0 0 0' equivalent to *enabled*='false'

Common fields: *enterTime, exitTime*

enterTime and *exitTime* are SFTime output values sent whenever the sensor proximity or visibility condition is met

- These timestamp values are helpful triggers for other animation chains

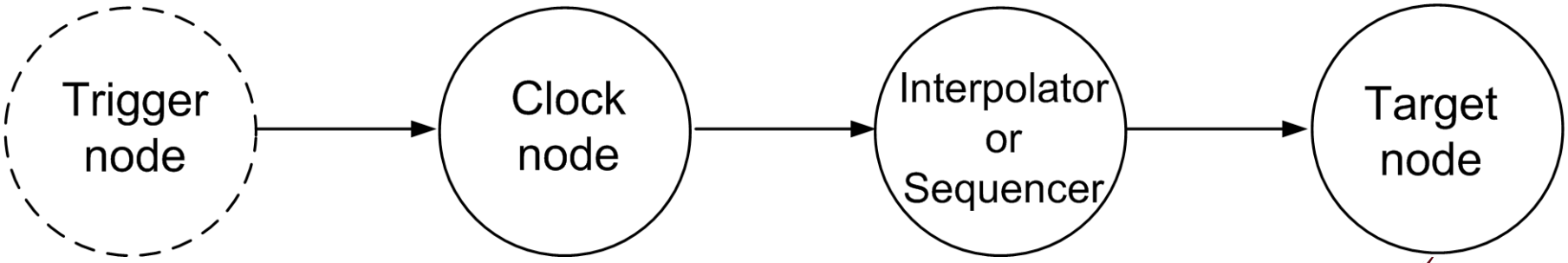
Corresponding *isActive* event sent

- true or false, corresponds to *enterTime* or *exitTime*
- accompanying timestamp values also matches

review

Example behavior event chain

- User clicks button to start a timer clock
- Clock outputs new event at start of each frame,
- ... which stimulates linear-interpolation function which produces another output value
- ... which updates some target value in scene graph
- Repeat event traversal after each frame redraw



X3D Nodes and Examples

LoadSensor node

LoadSensor keeps track of progress & completion when downloading external file resources

- AudioClip, ImageTexture, Inline, MovieTexture, etc.
- Can't track Anchor (viewpoint shift or external link)
- These child nodes have *containerField*='watchList'

Fields include *enabled*, *watchList* nodes, *timeOut, progress*, *isActive*, *isLoaded, loadTime*

- *enabled* identical to usage in other sensor nodes

Helpful as trigger node in event-animation chain

- Can delay animations until all resources available

LoadSensor fields: *watchList*, *timeOut*

watchList is an MFNode array that lists the nodes of interest to be monitored by LoadSensor

- Usually USE copies of DEF nodes appearing elsewhere in scene
- Each watched node must have *url* field list
- All nodes listed in the *watchList* array must succeed for LoadSensor to succeed

timeOut defines maximum seconds to wait

- default 0 means indefinite, no time limit
- External network source might nevertheless time out if excessive time required

LoadSensor fields: *progress*, *isLoaded*, *loadTime*

progress is an SFFloat output with value [0..1]

- Only reaches value of 1 when complete
- Browsers decide how often to issue output values
- Browsers also decide on precise meaning: file percentage, download time, bytes downloaded, etc.

isActive reports true/false when actively loading

isLoaded sends a true event when complete OK, otherwise sends false when *isActive* goes false

loadTime also sent, having the same timestamp, when the *isLoaded*='true' event is sent

LoadSensor hints

Good design practices:

- Include LoadSensor as trigger in animation chain if target animation makes no sense without resources
- Plan on having a default behavior or warning, just in case the resources don't load

Use multiple LoadSensor nodes for multiple resources, if precise progress results needed

- Can't detect which address in *url* array succeeds
- Can combine with Script to sequence addresses

LoadSensor warnings

All child nodes within LoadSensor must have *containerField*='watchList'

- Or else a run-time error results

LoadSensor not usable for ExternProtoDeclare, because ExternProtoDeclare not a node, per se

- How: embed a Script inside the ProtoDeclare, then have ProtoInstance initialize() method send an output event once initialization is complete
- Can also embed LoadSensor inside ProtoDeclare, then expose relevant fields using IS/connect

LoadSensor.x3d - Editor

LoadSensor.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.2//EN" "http://www.web3d.org/specifications/x3d-3.2.dtd">
<X3D profile='Immersive' version='3.2' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocati
    <head>
        <meta content='LoadSensor.x3d' name='title'/>
        <meta content='Simple test of LoadSensor node' name='description'/>
        <meta content='Don Brutzman' name='creator'/>
        <meta content='17 July 2008' name='created'/>
        <meta content='17 July 2008' name='modified'/>
        <meta content='Copyright Don Brutzman and Leonard Daly 2008' name='rights'/>
        <meta content='X3D LoadSensor example' name='subject'/>
        <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/LoadSensor.x3d'
        <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
        <meta content='../../license.html' name='license'/>
    </head>
    <Scene>
        <Viewpoint DEF='ViewAll' position='0 0 12'/>
        <Inline DEF='HelloWorld' url='"../HelloWorld.x3d" "http://www.web3d.org/x3d/content/examples/HelloWorld.x3d"'/>
        <LoadSensor DEF='InlineLoadSensor'>
            <Inline USE='HelloWorld' containerField='watchList'/>
        </LoadSensor>
        <Transform DEF='PivotTextSigns' translation='0 3 0'>
            <!-- First sign indicates waiting... -->
            <Shape>
            <!-- Second sign indicates completed, initially rotated 90 degrees out of view -->
            <Transform>
        </Transform>
        <OrientationInterpolator DEF='SignRotationInterpolator' key='0 1' keyValue='0 1 0 0, 0 1 0 -1.57'/>
        <TimeSensor DEF='SignRotationClock' cycleInterval='0.8'/>
        <ROUTE fromField='loadTime' fromNode='InlineLoadSensor' toField='startTime' toNode='SignRotationClock'/>
        <ROUTE fromField='fraction_changed' fromNode='SignRotationClock' toField='set_fraction' toNode='SignRotationInterpolator'/>
        <ROUTE fromField='value_changed' fromNode='SignRotationInterpolator' toField='rotation' toNode='PivotTextSigns'/>
        <Script DEF='TraceScript'>
            <field accessType='inputOnly' name='isLoadedEvent' type='SFBool'/>
<![CDATA[
ecmascript:
function isLoadedEvent (value)
{
        Browser.println ('InlineLoadSensor isLoaded=' + value);
}]]>
        </Script>
        <ROUTE fromField='isLoaded' fromNode='InlineLoadSensor' toField='isLoadedEvent' toNode='TraceScript'/>
    </Scene>
</X3D>
```

output to console

Edit LoadSensor

containerField

children

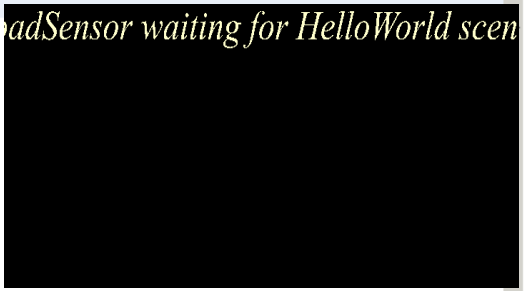
DEF InlineLoadSensor

USE

enabled

timeOut 5

OK Cancel Help



19:20 INS

LoadSensor	**LoadSensor generates events as watchList child nodes are either loaded or fail to load. Changing watchlist child nodes restarts the LoadSensor.** **Hint: use multiple LoadSensor nodes to track multiple loading nodes individually.** **Hint: Background is not sensed due to multiple-image ambiguity.** **Warning: watchList child nodes are not rendered, so normally USE copies of other nodes to sense load status.** **Hint: use Inline 'load' field to prompt or defer loading.** **Warning: new X3D node, not supported in VRML 97.**
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!
enabled	**[enabled: accessType inputOutput, type SFBool (true|false) "true"]** Enables/disables node operation.
timeOut	**[timeOut: accessType inputOutput, type SFTime CDATA "0" #IMPLIED]** Time in seconds of maximum load duration prior to declaring failure. Default value zero means use browser defaults.
isActive	**[isActive: outputOnlytype SFBool (true|false) #FIXED ""]** isActive true/false events are sent when load sensing starts/stops.
isLoaded	**[isLoaded: accessType outputOnly, type SFBool (true|false) #FIXED ""]** Notify when all watchList child nodes are loaded, or at least one has failed. Sends true on successfully loading all watchList child nodes. Sends false on timeOut of any watchList child nodes, failure of any watchList child nodes to load, or no local copies available and no network present. Hint: use multiple LoadSensor nodes to track multiple loading nodes individually.
loadTime	**[loadTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time of successful load complete, not sent on failure.
progress	**[progress: accessType outputOnly, type SFFloat CDATA [0.0 .. 1.0]** #FIXED ""] Sends 0.0 on start and 1.0 on completion. Intermediate values are browser dependent and always increasing (may indicate fraction of bytes, fraction of expected time or another metric). Hint: only 0 and 1 events are guaranteed.
containerField	**[containerField: NMTOKEN "children"]** 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.

ProximitySensor node

ProximitySensor tracks user position, orientation within box defined by author as active volume

- Tracking box is not visibly rendered
- Tracking box is affected by parent transformations

Can track (and react to) user navigation in scene

- Able to send multiple events: *position_changed*, *orientation_changed*, *centerOfRotation_changed*

Fields also include *center*, *size*, *enabled*, *isActive*, *enterTime*, *exitTime*

Helpful as trigger node in event-animation chain

- Can delay animations until user is close to action

Output events

position_changed reports location relative to ProximitySensor *center*

- Best is to keep *center* at origin, with very large *size*

orientation_changed events sent when user's view rotates within the monitored volume

- Also occurs if proximity box rotates independently

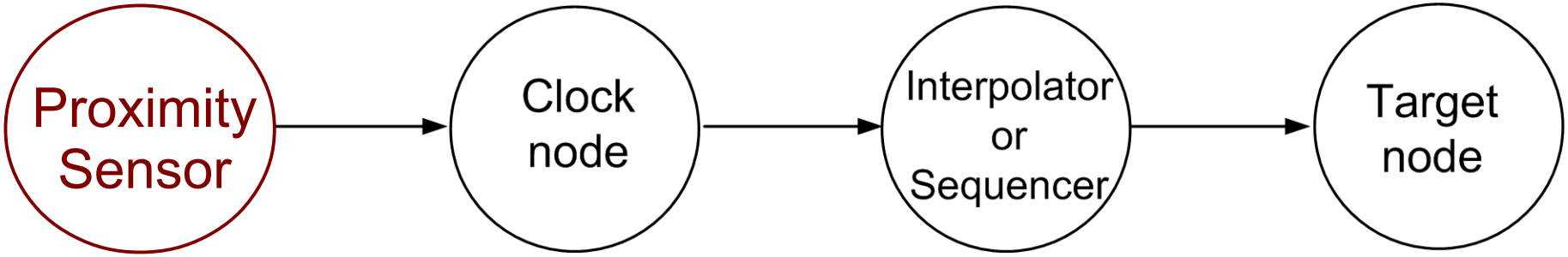
centerOfRotation_changed monitors output of NavigationInfo if LOOKAT point changes

- Viewer must be in LOOKAT mode for this to occur

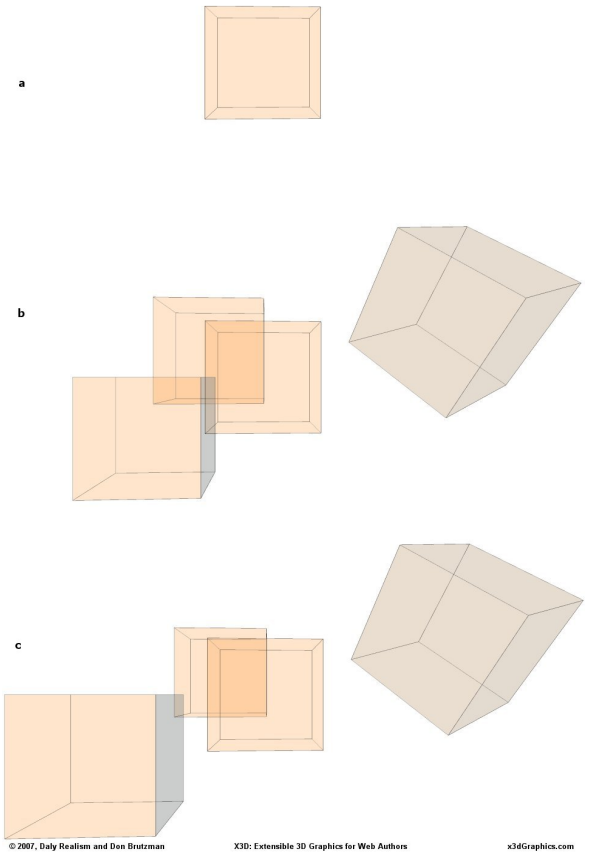
Example usage: animation trigger

ProximitySensor can start an animation, running only when results are close enough to be seen

- Similarly turn animation off when no longer nearby
- Reduces computational cost, enabling larger scenes
- Thus helps eliminate off-screen animation, where unseen event chains might spin needlessly



Example ProximitySensor sensing volumes



a. single instance

b. multiple independent nonoverlapping instances

c. single DEF ProximitySensor with multiple transformed nonoverlapping USE instances

web3D CONSORTIUM

ProximitySensor.x3d - Editor

ProximitySensor.x3d

```
<!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/specificat
  <head>
    <meta content='ProximitySensor.x3d' name='title'/>
    <meta content='Demonstrates use of a ProximitySensor in building a HUD' name='descript
    <meta content='Leonard Daly and Don Brutzman' name='creator'/>
    <meta content='15 July 2006' name='created'/>
    <meta content='27 December 2007' name='modified'/>
    <meta content='http://X3dGraphics.com' name='reference'/>
    <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference'/>
    <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
    <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='sub
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentS
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
  </head>
  <Scene>
    <ProximitySensor DEF='HereIAm' enabled='true' size='1000 1000 1000'/>
    <Transform DEF='HUD'>
      <Transform DEF='PushBack' translation='-.8 1. -3'>
        <Shape>
        <Transform DEF='HudContents' translation='-.7 .3 .1'>
          <Transform DEF='TopText' translation='0 -.2 0'>
            <Shape>
              <Appearance DEF='Text_app'>
                <Material diffuseColor='0 0 0'/>
              </Appearance>
              <Text DEF='TopTextLine' string='"Position 0.00 0.00 10.00"'>
                <FontStyle DEF='TextStyle' family='"SANS SERIF"' justify='"BEGIN"' size='.1' style='PLAIN'/>
              </Text>
            </Shape>
          </Transform>
          <Transform>
        </Transform>
      </Transform>
    </Transform>
    <Script DEF='CnvText' url='"convertText.js" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-Envi
      <field accessType='inputOnly' name='position' type='SFVec3f'/>
      <field accessType='inputOnly' name='orientation' type='SFRotation'/>
      <field accessType='outputOnly' name='positionText' type='MFString'/>
      <field accessType='outputOnly' name='orientationText' type='MFString'/>
    </Script>
    <ROUTE fromField='orientation_changed' fromNode='HereIAm' toField='rotation' toNode='HUD'/>
    <ROUTE fromField='position_changed' fromNode='HereIAm' toField='translation' toNode='HUD'/>
    <ROUTE fromField='position_changed' fromNode='HereIAm' toField='position' toNode='CnvText'/>
    <ROUTE fromField='orientation_changed' fromNode='HereIAm' toField='orientation' toNode='CnvText'/>
    <ROUTE fromField='positionText' fromNode='CnvText' toField='string' toNode='TopTextLine'/>
    <ROUTE fromField='orientationText' fromNode='CnvText' toField='string' toNode='BottomTextLine'/>
    <Inline url='"../KelpForestExhibit/KelpForestMain.x3d" "../KelpForestExhibit/KelpForestMain.wrl" "http://X3dGraphics.com/examples/X3dForWebAuthors/Kelp
```

38:11 INS

Xj3D Browser - Aviatrix3D

File View Viewpoint Navigation Options Help

Location: rg/x3d/content/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/ProximitySensor.x3d

Position 42.32 14.08 7.82

Orientation -0.34 0.94 -0.031 1.4

Kelp Forest Monterey Bay Aquarium

File 'C:/www.web3d.org/x3d/content/examples/X3dForWebAuthors/KelpForestExhibit/SharkDarkEye.jpg' read

32.25

convertText.js - Editor

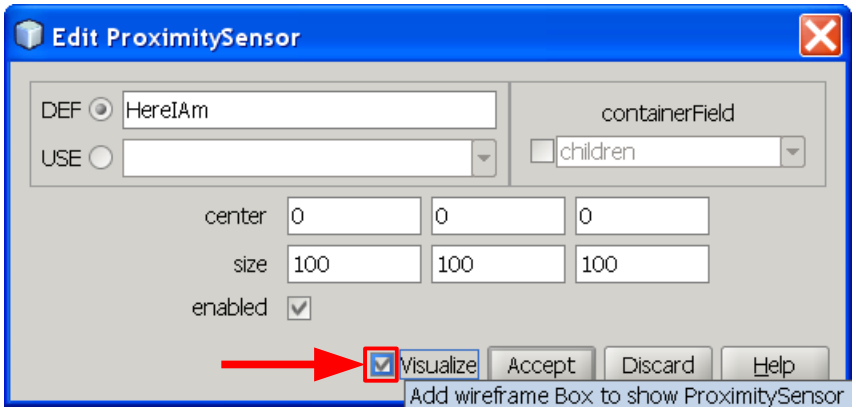
convertText.js

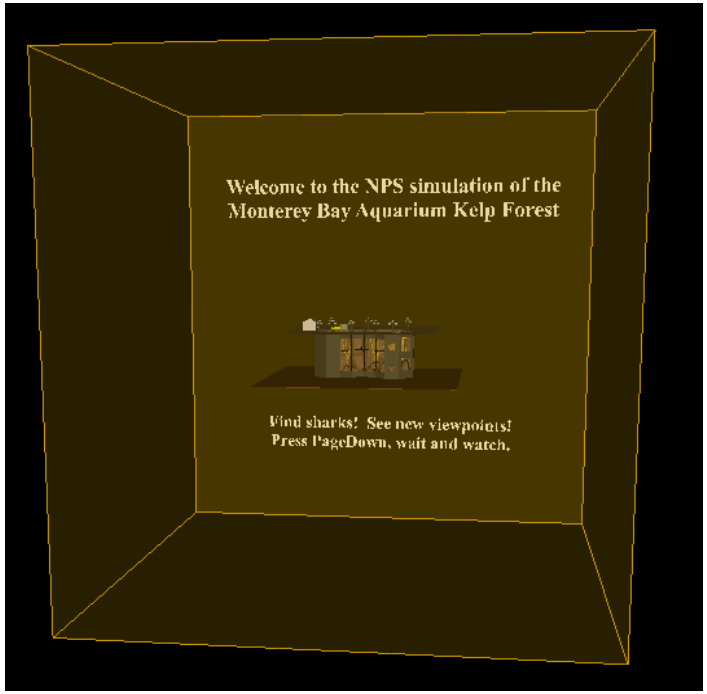
```
ecmascript:

function position (value) {
  x = setDigits (value[0], 100);
  y = setDigits (value[1], 100);
  z = setDigits (value[2], 100);

  positionText = new MFString ('Position  ' +
     x + '  ' + y + '  ' + z);
//  Browser.print (positionText[0]+'\n');
}
function orientation (value) {
  x = setDigits (value[0], 1000);
  y = setDigits (value[1], 1000);
  z = setDigits (value[2], 1000);
  r = setDigits (value[3], 100);
  orientationText = new MFString ('Orientation  ' +
     x + '  ' + y + '  ' + z + '  ' + r);
//  Browser.print (orientationText[0]+'\n');
}

function setDigits (v, p) {
  return Math.floor (v*p + .5) / p;
}
```

9:7 INS

ProximitySensor visualization

- Visualization option provided by X3D-Edit
  - Displays transparent wireframe box outlining ProximitySensor *size* boundaries
  - Also displays coordinate axes at *center* location
  - `<ProximitySensor DEF='HereIAm' size='100 100 100'/>`





ProximitySensor	**ProximitySensor generates events when the viewer enters, exits and moves within a region of space (defined by a box).** **Hint: multiple USEd instances are cumulative, but avoid overlaps.** **Hint: can first use Transform to relocate/reorient box.** **Hint: surround entire world to start behaviors once scene is loaded.**
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!
enabled	**[enabled: accessType inputOutput, type SFBool (true|false) "true"]** Enables/disables node operation.
center	**[center: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** Position offset from origin of local coordinate system.
size	**[size: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** size of Proximity box. **Hint:** size 0 0 0 is same as enabled false.
isActive	**[isActive: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isActive true/false events are sent as viewer enters/exits Proximity box. isActive=true when viewer enters Proximity box, isActive=false when viewer exits Proximity box.
position_changed	**[position_changed: accessType outputOnly, type SFVec3f CDATA #FIXED ""]** Sends translation event relative to center.
orientation_changed	**[orientation_changed: accessType outputOnly, type SFRotation CDATA #FIXED ""]** Sends rotation event relative to center.
centerOfRotation_changed	**[centerOfRotation_changed: accessType outputOnly, type SFRotation CDATA #FIXED ""]** Sends changed centerOfRotation values, likely caused by user interaction.
enterTime	**[enterTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera enters the box.
exitTime	**[exitTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera exits the box.
containerField	**[containerField: NMTOKEN "children"]** 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.

VisibilitySensor node

VisibilitySensor detects whether a given volume of space is visible from current user view

- Depends on user's current direction and position
- Not dependent on proximity distance or range rate
- Intermediate occluding geometry has no effect

Fields include *center*, *size*, *enabled*, *isActive*, *enterTime*, *exitTime*

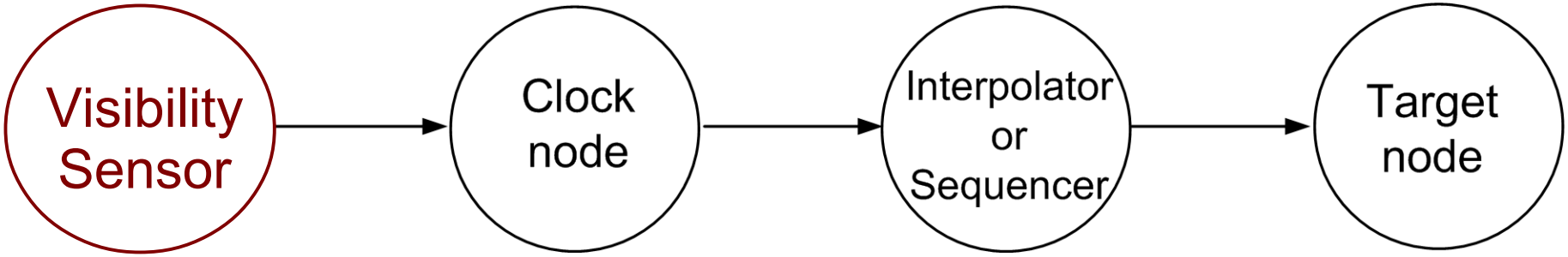
Helpful as trigger node in event-animation chain

- Can delay animations until user is able and ready to view them

Example usage: animation trigger

VisibilitySensor can control animation, only running when results are visible

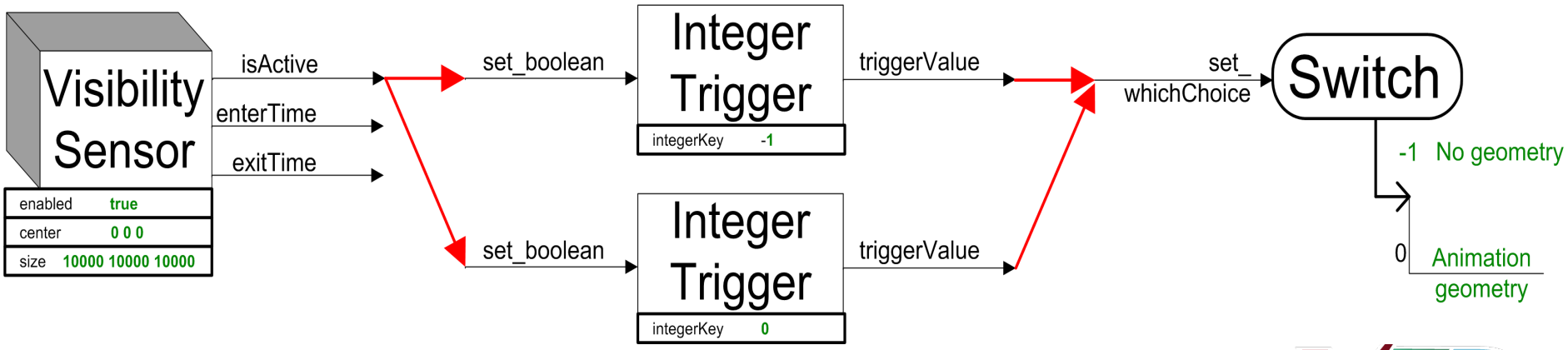
- Similarly turn animation off when no longer visible
- Reduces computational cost, enabling larger scenes



Example usage: switch out geometry

VisibilitySensor can remove scene subgraphs, only showing them when results are visible

- Similarly removing them when no longer visible
- Reduces computational cost, enabling larger scenes



VisibilitySensor-KelpForestMain.x3d - Editor

VisibilitySensor-KelpForestMain.x3d

```
<?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:noNamespaceSc
  <head>
    <meta content='VisibilitySensor-KelpForestMain.x3d' name='title'/>
    <meta content='VisibilitySensor example.' name='description'/>
    <meta content='Don Brutzman Revised: Leonard Daly for X3D Book' name='creator'/>
    <meta content='1 June 1998' name='created'/>
    <meta content='8 October 2007' name='modified'/>
    <meta content='http://web.nps.navy.mil/~brutzman/kelp' name='reference'/>
    <meta content='http://web.nps.navy.mil/~brutzman/kelp/KelpForestDesignPaper.pdf' name='reference'/>
    <meta content='Kelp Forest 3D models' name='subject'/>
    <meta content='All content has permissions for free use. Please provide credit to the Naval Postgraduate Sc
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/KelpForestExhibit/KelpForestMain.x3d' name=
    <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/Visibilit
    <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
    <meta content='../license.html' name='license'/>
  </head>
  <Scene>
    <Collision enabled='false'>
      <Transform translation='0 -3.5 -2'>
        <Shape>
          <Appearance>
            <Material diffuseColor='1 .5 0' transparency='.9'/>
          </Appearance>
          <Box size='20 11 11'/>
        </Shape>
      </Transform>
    </Collision>
    <VisibilitySensor DEF='SeeMe' center='0 -3.5 -2' enabled='true' size='20 11 11'/>
    <Script DEF='PrintInfo' url='"environmentalSensors.js" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/environm
      <field accessType='inputOnly' name='isVisible' type='SFBool'/>
    </Script>
    <ROUTE fromField='isActive' fromNode='SeeMe' toField='isVisible' toNode='PrintInfo'/>
    <WorldInfo info='"../KelpForest/kelpForestMain3.4.wrl" "Model of the Monterey Bay Aquarium Kelp Forest" "DTG of last updated: 111500Jun98" "Added:
    <NavigationInfo speed='3.0'/>
    <ExternProtoDeclare name='ViewPositionOrientation' url='"../KelpForest/ViewPositionOrientationPrototype.wrl#ViewPositionOrientation" "/www.web3d.c
      <field accessType='inputOutput' name='enabled' type='SFBool'/>
      <field accessType='initializeOnly' name='traceEnabled' type='SFBool'/>
      <field accessType='inputOnly' name='set_traceEnabled' type='SFBool'/>
      <field accessType='outputOnly' name='position_changed' type='SFVec3f'/>
      <field accessType='outputOnly' name='orientation_changed' type='SFRotation'/>
      <field accessType='outputOnly' name='outputViewpointString' type='MFString'/>
    </ExternProtoDeclare>
    <ProtoInstance name='ViewPositionOrientation'>
      <fieldValue name='enabled' value='false'/>
    </ProtoInstance>
```

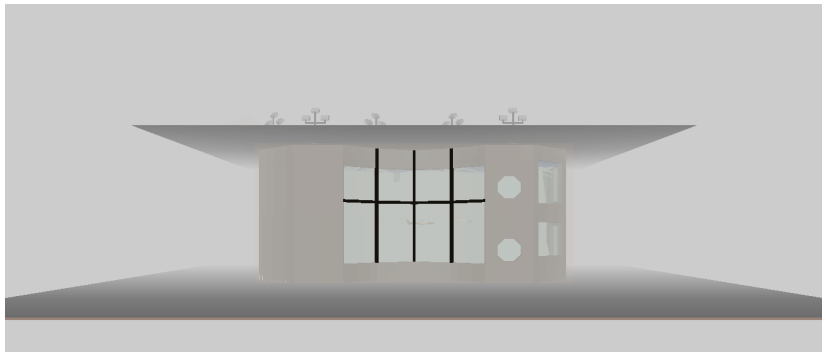
30:22 INS

environmentalSensors.js - Editor

environmentalSensors.js

```
// Functions for ProximitySensor

function position (value) {
  x = Math.floor (value[0] * precision + .5) / precision;
  y = Math.floor (value[1] * precision + .5) / precision;
  z = Math.floor (value[2] * precision + .5) / precision;
  print ('position: ' + x + ' ' + y + ' ' + z);
}

function orientation (value) {
  x = Math.floor (value[0] * precision + .5) / precision;
  y = Math.floor (value[1] * precision + .5) / precision;
  z = Math.floor (value[2] * precision + .5) / precision;
  a = Math.floor (value[3] * precision + .5) / precision;
  print ('orientation: ' + x + ' ' + y + ' ' + z + ' ' + a);
}

// Functions for VisibilitySensor

function isVisible (value) {
  print ('The region is visible: ' + value);
}
```

21:32 INS



Edit VisibilitySensor

containerField: children

DEF: SeeMe

USE:

center	0	-3.5	-2
size	20	11	11

enabled: ☑

OK Cancel Help

VisibilitySensor	**VisibilitySensor detects when user can see a specific object or region as they navigate the world. This region is bounded by a box.** **Hint: often used to attract user attention or improve performance.** **Hint: Sensors are affected by peer nodes and children of peers.**
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!
enabled	**[enabled: accessType inputOutput, type SFBool (true|false) "true"]** Enables/disables node operation.
center	**[center: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** Translation offset from origin of local coordinate system.
size	**[size: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** size of visibility box, measured from center in meters.
isActive	**[isActive: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isActive true/false events are sent when triggering the sensor. isActive=true when entering visibility region, isActive=false when exiting visibility region.
enterTime	**[enterTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera enters visibility region for sensor.
exitTime	**[exitTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera exits visibility region for sensor.
containerField	**[containerField: NMTOKEN "children"]** 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.

Sound node

Sound node specifies source, location, intensity, direction, and spatial characteristics of each sound source in the scene

Source audio for the sound to be played is provided by a child AudioClip node

- Or alternatively by child MovieTexture soundtrack

Multiple stationary and moving sounds can be rendered together with geometry, improving realism and liveness within animated scenes

Sound fields: *location, direction, intensity, priority*

location is center position of sound origin

- Relative to local coordinate system

direction is unit-vector direction of sound axis

- Three-tuple vector, not a four-tuple SFRotation
- Relative to local coordinate system

intensity is factor [0..1] adjusts loudness of emitted sound

- Multiplied against original volume level of source

priority [0..1] is a browser hint, if ever needed, to choose which of several sounds to play

Sound fields: min/max Front/Back

minFront is minimum-attenuation (full volume) ellipsoid distance, along *direction* axis

maxFront is maximum-attenuation (zero volume) ellipsoid distance, along *direction* axis

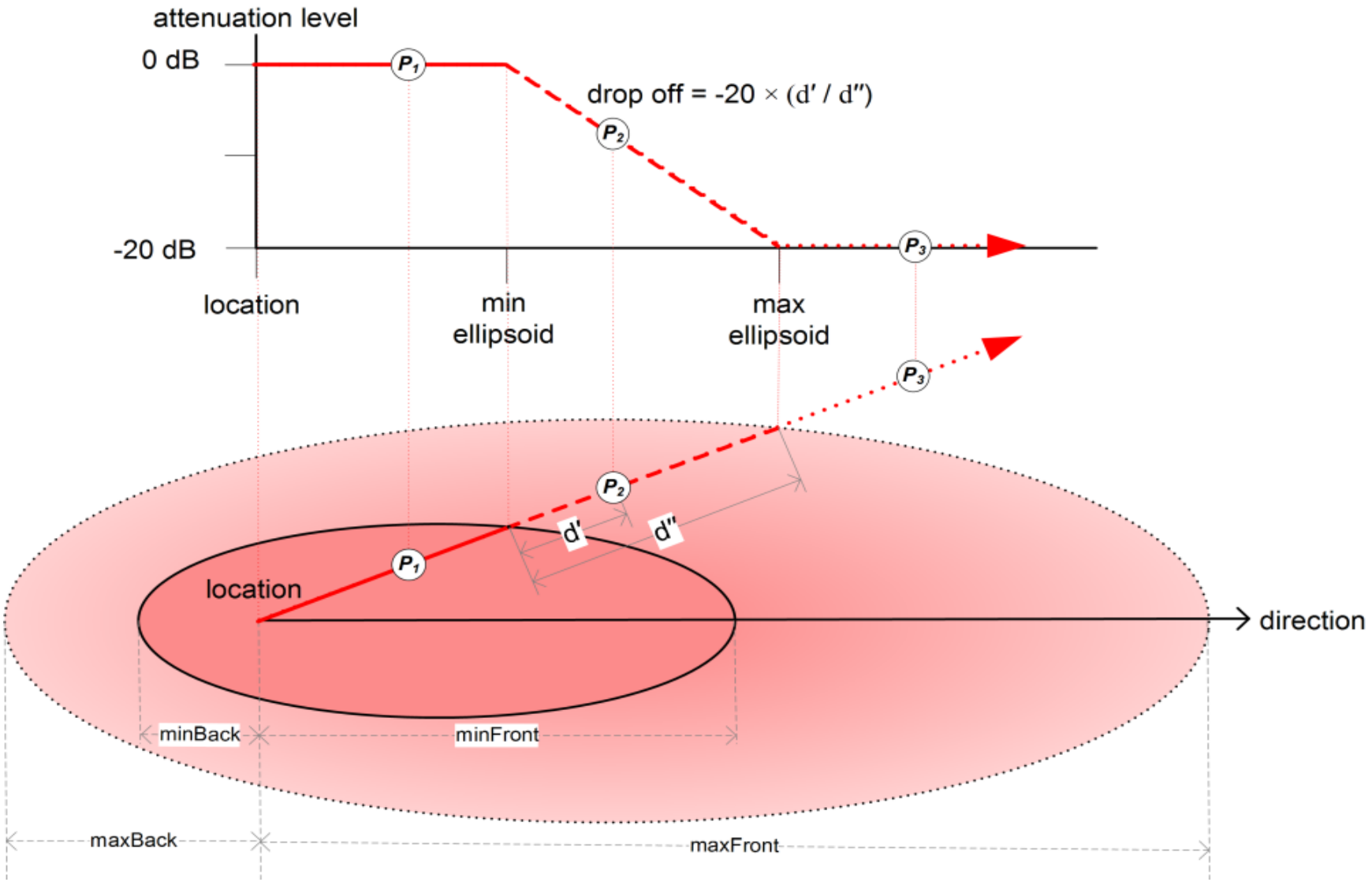
- Must ensure *minFront* <= *maxFront*

minBack is minimum-attenuation (full volume) ellipsoid distance, opposite *direction* axis

maxBack is maximum-attenuation (zero volume) ellipsoid distance, opposite *direction* axis

- Must ensure *minBack* <= *maxBack*

Sound ellipses for front, back boundaries



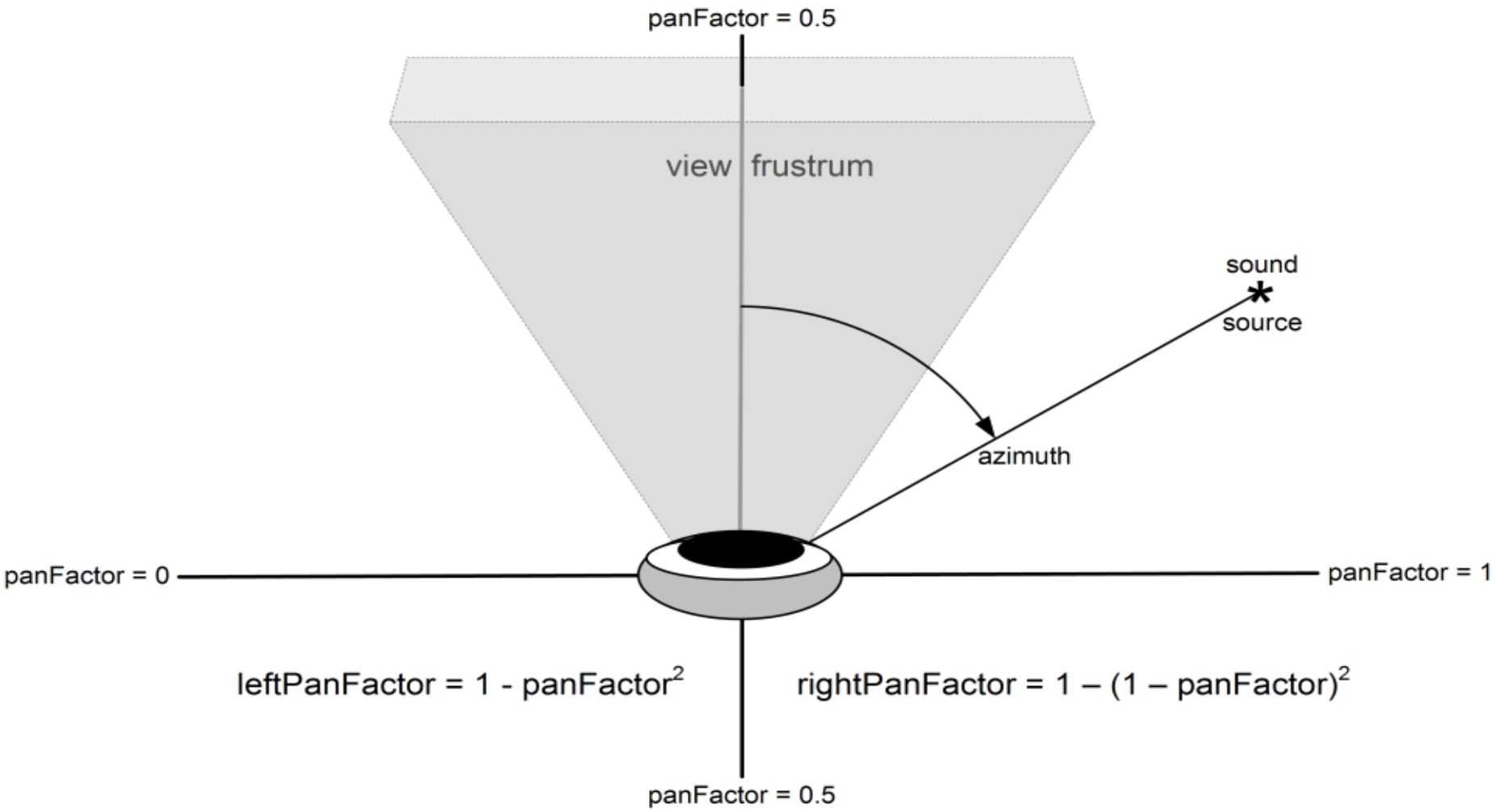
Sound field: *spatialize*

spatialize is a boolean determining whether to spatialize sound playback relative to viewer

- Only acts between minimum, maximum ellipsoids
- Stereo effect

Pan-factor computations determine spatialization effect on each channel

Stereo left-right pan factors



Computing ellipsoid dimensions

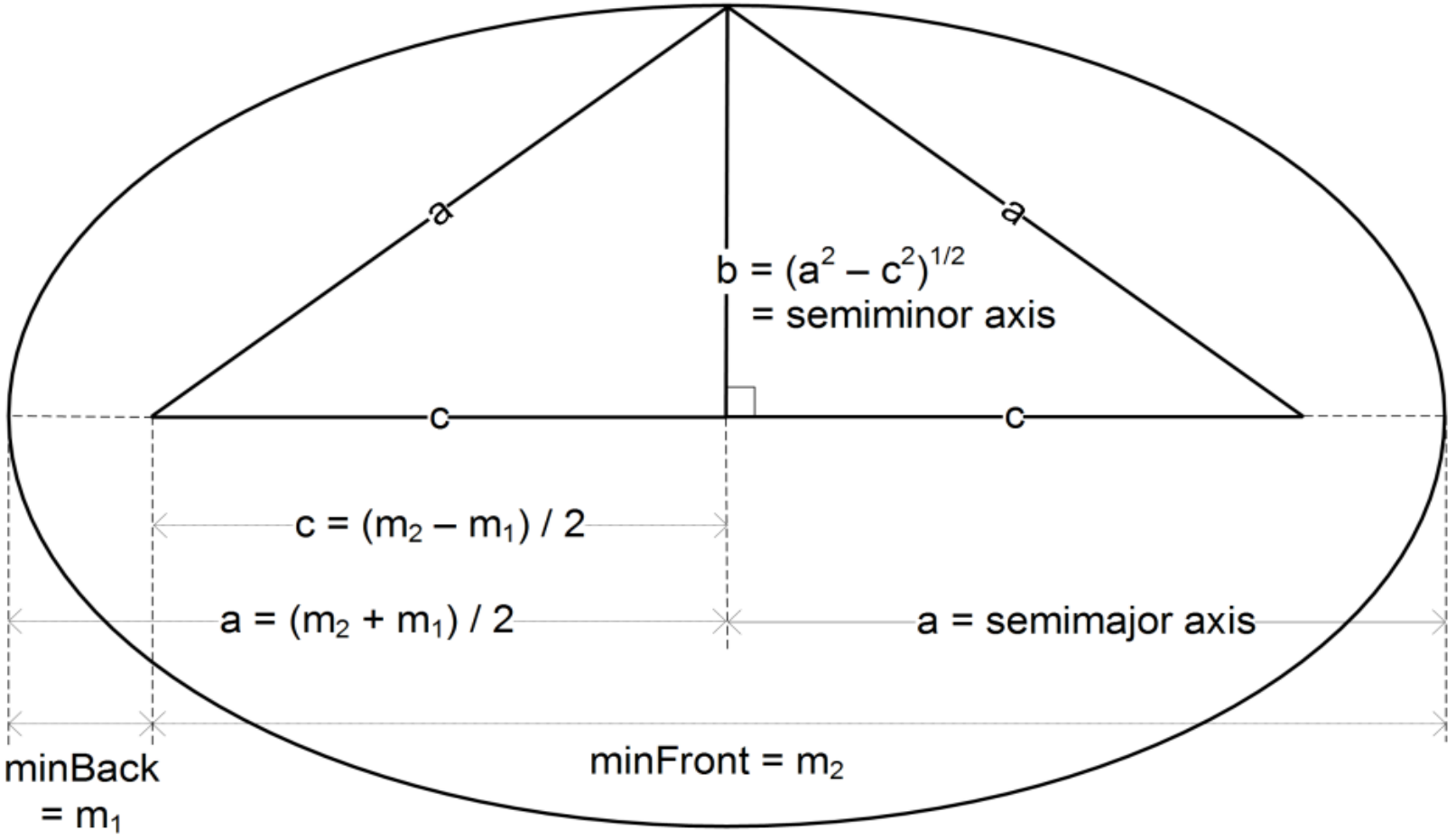
location is focus point of the sound ellipsoid

maxFront, *maxBack* define dimensions of outer ellipsoid along and opposite the *direction* axis

Note however that ellipsoid width is not defined by these parameters, because that value is dependent on the other definitions

- The next diagram, equations show how to calculate

Ellipse major, minor axes



Example values for this diagram:

minBack=1 and minFront=9 produce ellipse a=5, b=3, c=4

Calculating the width of sound ellipsoid

$$minHalfWidth = b_{\min} = \sqrt{a^2 - c^2}$$

$$= \sqrt{\left[\frac{m_1 + m_2}{2}\right]^2 - \left[\frac{m_1 - m_2}{2}\right]^2}$$

$$= \frac{1}{2}\sqrt{m_1^2 + 2m_1m_2 + m_2^2 - \left(m_1^2 - 2m_1m_2 + m_2^2\right)}$$

$$= \frac{1}{2}\sqrt{2m_1m_2 + 2m_1m_2}$$

$$= \frac{1}{2}(2)\sqrt{m_1m_2} = \sqrt{m_1m_2}$$

$$minHalfWidth = \sqrt{minBack \cdot minFront}$$

similarly

$$maxHalfWidth = \sqrt{maxBack \cdot maxFront}$$

SoundAudioClip.x3d - Editor

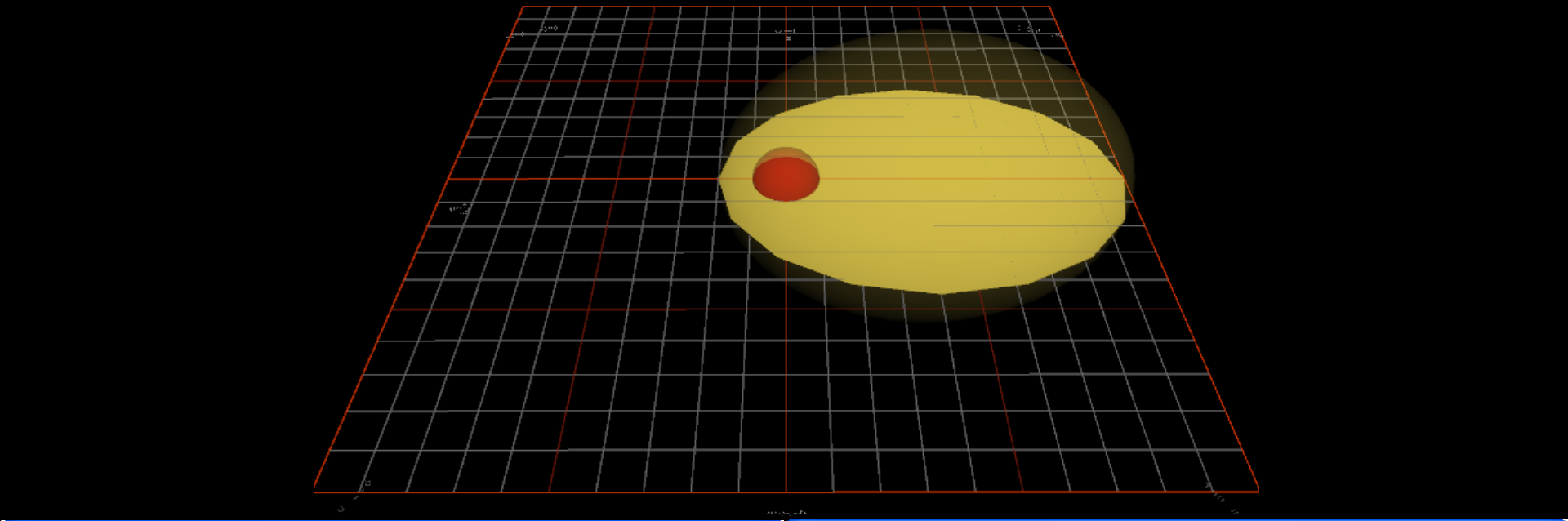
SoundAudioClip.x3d

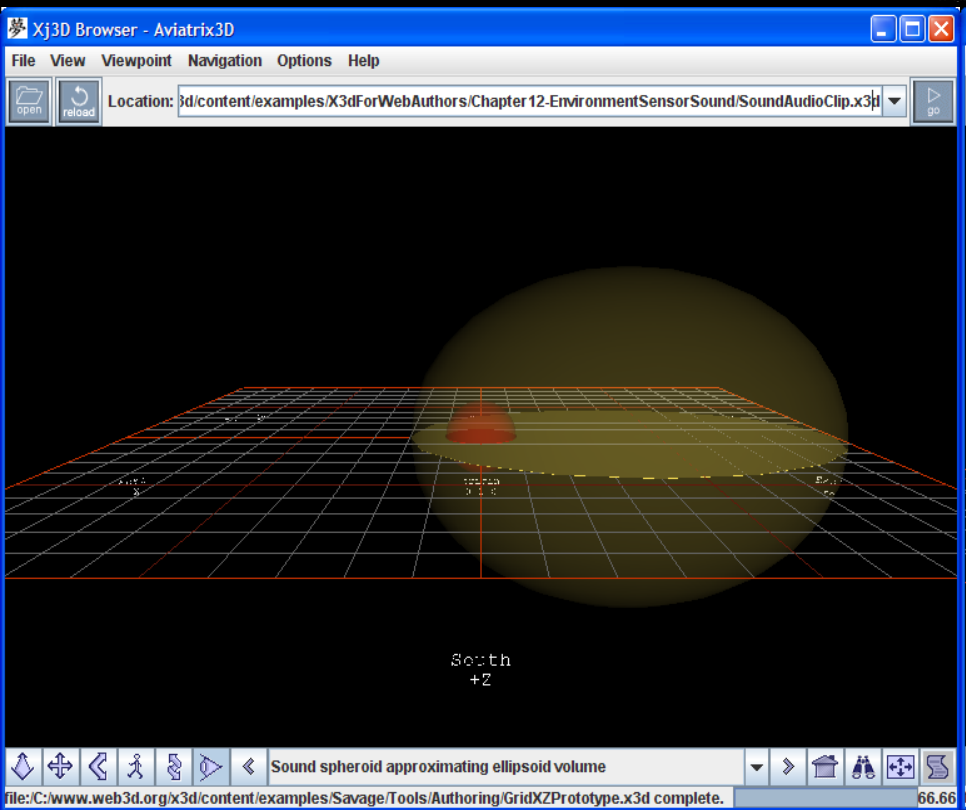
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.1//EN" "http://www.web3d.org/specifications/x3d-3.1.dtd">
<X3D profile='Immersive' version='3.1' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchemaLocation='http://www.web3d.org/speci
    <head>
        <meta content='SoundAudioClip.x3d' name='title'/>
        <meta content='An example of the Sound and AudioClip node showing the effect of the various volume regions' name='description'/>
        <meta content='1 May 2006' name='created'/>
        <meta content='20 July 2008' name='modified'/>
        <meta content='http://X3dGraphics.com' name='reference'/>
        <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference'/>
        <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
        <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
        <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/SoundAudioClip.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='Sound spheroid approximating ellipsoid volume' position='0 40 200' orientation='1 0 0 -0.2'/>
        <Sound DEF='Audible' direction='1 0 0' maxBack='20' maxFront='100' minBack='10' minFront='10' priority='1'>
            <AudioClip DEF='WaterSounds' description='Running Water' url='"aqua.wav" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-Envir
        </Sound>
        <!-- minWidth = sqrt (minBack * minFront) = 10 -->
        <!-- maxWidth = sqrt (maxBack * maxFront) = 45 -->
        <!-- Approximate ellipsoidal footprints and envelope using Cylinders and Spheres, respectively -->
        <!-- Minimum attenuation parameters 10, 10, 10 produce a Cylinder and Sphere with uniform scale
        <Transform DEF='minShapes' scale='10 10 10'>
            <Shape>
                <Cylinder height='0.05' />
                <Appearance>
                    <Material DEF='MinFootprintMaterial' diffuseColor='0.8 0 0' />
                </Appearance>
            </Shape>
            <Shape>
                <Sphere/>
                <Appearance>
                    <Material DEF='MinSpheroidMaterial' diffuseColor='0.8 0 0' transparency='0.5'/>
                </Appearance>
            </Shape>
        </Transform>
        <!-- Maximum attenuation parameters 20, 100, 45 produce a Cylinder and Sphere with non-uniform scale -->
        <!-- Sound direction is x axis, so maximum-attenuation length = (100 + 20) = 120 and radius = 120 / 2 = 60 along x axis -->
        <!-- and so width, height dimensions ~= maximum-attenuation halfwidth = maxWidth ~= 45 along y, z axes -->
        <!-- Ellipsoid focus is (0 0 0) but center of spheroidal approximation is (-maxBack + x radius) = (-20 + 60) = 40 along x axis -->
        <Transform DEF='maxShapes' translation='40 0 0' scale='60 45 45'>
            <Shape>
                <Cylinder height='0.002' />
```

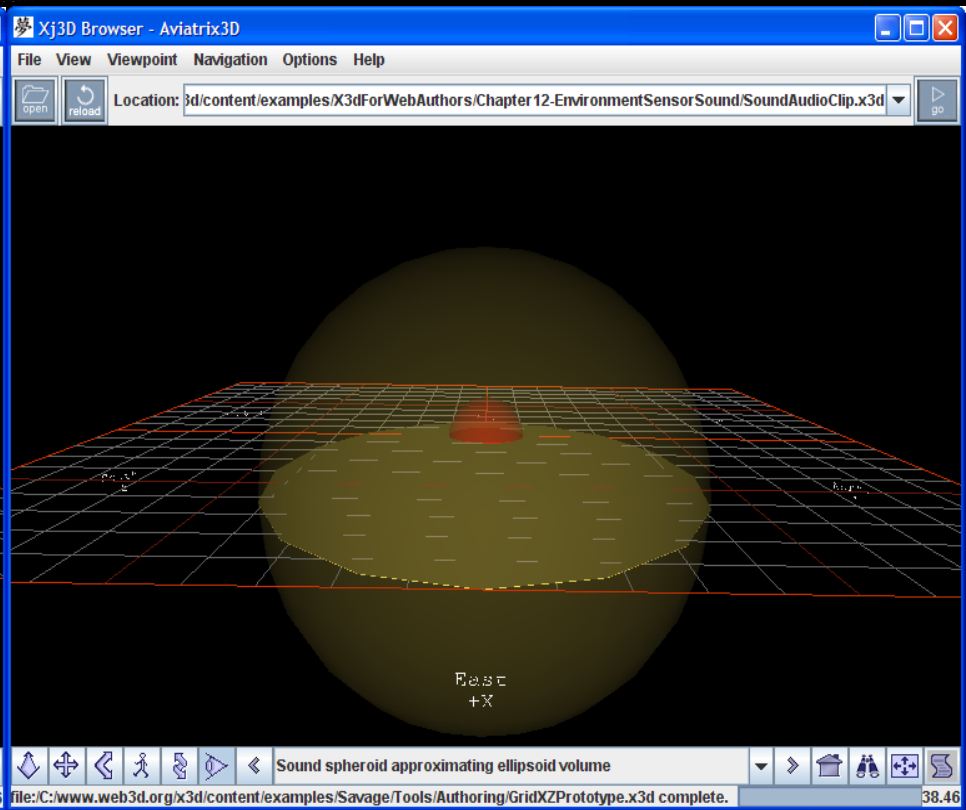
Edit Sound

DEF Audible
USE
containerField
children
location 0 0 0
direction 1 0 0
intensity 1
priority 1
spatialize
minBack 10 10 minFront
maxBack 20 100 maxFront
show outline
Accept Discard Help

19:15 INS




Xj3D Browser - Aviatrix3D
File View Viewpoint Navigation Options Help
open reload Location: 3d/content/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/SoundAudioClip.x3d go
South
+Z
Sound spheroid approximating ellipsoid volume
file:/C:/www.web3d.org/x3d/content/examples/Savage/Tools/Authoring/GridXZPrototype.x3d complete.
66.66


Xj3D Browser - Aviatrix3D
File View Viewpoint Navigation Options Help
open reload Location: 3d/content/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/SoundAudioClip.x3d go
East
+X
Sound spheroid approximating ellipsoid volume
file:/C:/www.web3d.org/x3d/content/examples/Savage/Tools/Authoring/GridXZPrototype.x3d complete.
38.46

Sound node hints

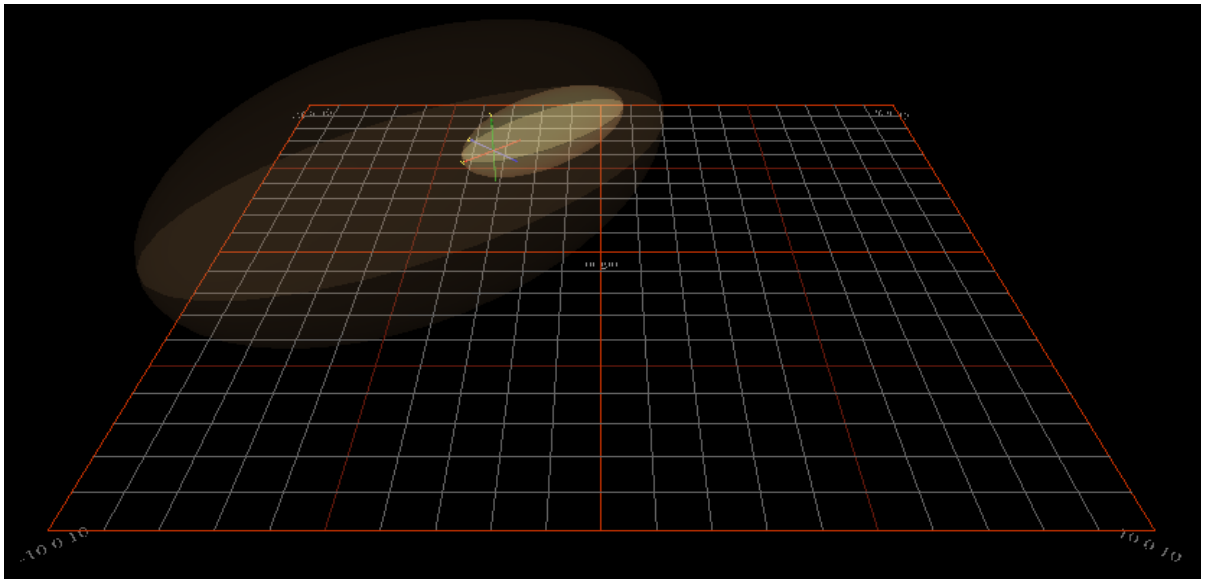
Since ellipses shrink quickly, raise up the Sound node's location to match Viewpoint eye level

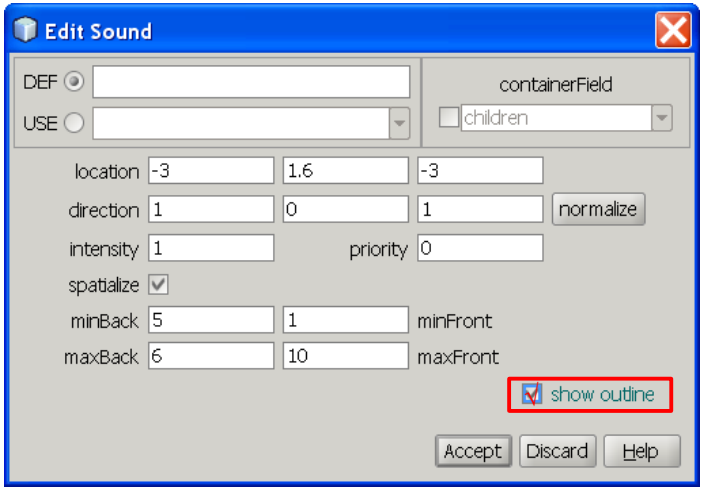
- When navigating in WALK mode, a typical *avatarSize* height is 1.6m
- Example: `<Sound location='0 1.6 0'/>`

Large volumes are good if you want to ensure the user hears what is intended

X3D-Edit includes a visualization author assist to transparently renders minimum (full volume) and maximum (zero audio volume) ellipsoids

Sound node visualization






```
<Sound location='-3 1.6 -3' direction='1 0 1' minBack='5' maxBack='6' minFront='1' maxFront='10'/>
```

X3D-Edit author-assist feature for Sound node automatically adds ellipsoid visualization to a scene

web3D CONSORTIUM

SoundAudioClip.x3d - Editor

SoundAudioClip.x3d

```
            <Shape>
                <Cylinder height='0.002' />
                <Appearance>
                    <Material DEF='MaxFootprintMaterial' diffuseColor='1 0.894118 0.341176'/>
                </Appearance>
            </Shape>
            <Shape>
                <Sphere/>
                <Appearance>
                    <Material DEF='MaxSpheroidMaterial' diffuseColor='1 0.894118 0.341176' transparency='0.75'/>
                </Appearance>
            </Shape>
        </Transform>
        <!-- Author-assist prototype from Savage X3D model archive -->
        <ExternProtoDeclare name='GridXZ' url=' "../../Savage/Tools/Authoring/GridXZPrototype.x3d#GridXZ"
            "../../../Savage/Tools/Authoring/GridXZPrototype.x3d#GridXZ" "https://savage.nps.edu/Savage/Tools/Authoring/GridXZPrototype.x3d#GridXZ" '>
            <field name='description' type='SFString' accessType='initializeOnly'/>
            <field name='labelColor' type='SFColor' accessType='inputOutput'/>
            <field name='scale' type='SFVec3f' accessType='inputOutput' appinfo='default unscaled size: 10m by 10m'/>
            <field name='labelsOffset' type='SFVec3f' accessType='inputOutput' appinfo='label location offset (in meters) to improve readability'/>
            <field name='originLabel' type='MFString' accessType='inputOutput'/>
            <field name='WestLabel' type='MFString' accessType='inputOutput' appinfo='-Z axis'/>
            <field name='NorthWestLabel' type='MFString' accessType='inputOutput'/>
            <field name='NorthLabel' type='MFString' accessType='inputOutput' appinfo='+X axis'/>
            <field name='NorthEastLabel' type='MFString' accessType='inputOutput'/>
            <field name='EastLabel' type='MFString' accessType='inputOutput' appinfo='+Z axis'/>
            <field name='SouthEastLabel' type='MFString' accessType='inputOutput'/>
            <field name='SouthLabel' type='MFString' accessType='inputOutput' appinfo='-X axis'/>
            <field name='SouthWestLabel' type='MFString' accessType='inputOutput'/>
        </ExternProtoDeclare>
        <!-- default values scaled up by factor of 10 in this instance -->
        <ProtoInstance name='GridXZ' containerField='children'>
            <fieldValue name='description' value='GridXZ'/>
            <fieldValue name='labelColor' value='1 1 1'/>
            <fieldValue name='scale' value='10 10 10'/>
            <fieldValue name='labelsOffset' value='0 -0.5 0'/>
            <fieldValue name='originLabel' value='"origin" "0 0 0"'/>
            <fieldValue name='NorthLabel' value='"North" "-Z"'/>
            <fieldValue name='NorthEastLabel' value='100 0 -100'/>
            <fieldValue name='EastLabel' value='"East" "+X"'/>
            <fieldValue name='SouthEastLabel' value='100 0 100'/>
            <fieldValue name='SouthLabel' value='"South" "+Z"'/>
            <fieldValue name='SouthWestLabel' value='-100 0 100'/>
            <fieldValue name='WestLabel' value='"West" "-X"'/>
            <fieldValue name='NorthWestLabel' value='-100 0 -100'/>
        </ProtoInstance>
    </Scene>
```

Visualization prototype from Savage X3D model archives superimposes an X-Z grid for easy 3D measurement.

58:67 INS

Sound	**Sound contains an AudioClip or MovieTexture for sound playback. 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!
location	**[location: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** Position of sound center, relative to local coordinate system.
direction	**[direction: accessType inputOutput, type SFVec3f CDATA "0 0 1"]** direction of sound axis, relative to local coordinate system.
intensity	**[intensity: accessType inputOutput, type SFFloat CDATA "1"]** Factor [0..1] adjusting loudness (decibels) of emitted sound.
minFront	**[minFront: accessType inputOutput, type SFFloat CDATA "1"]** Minimum-attenuation (full volume) ellipsoid distance, along direction ensure minFront <= maxFront.
minBack	**[minBack: accessType inputOutput, type SFFloat CDATA "1"]** Minimum-attenuation (full volume) ellipsoid distance, opposite direction ensure minBack <= maxBack.
maxFront	**[maxFront: accessType inputOutput, type SFFloat CDATA "10"]** Maximum-attenuation (zero volume) ellipsoid distance, along direction ensure minFront <= maxFront.
maxBack	**[maxBack: accessType inputOutput, type SFFloat CDATA "10"]** Maximum-attenuation (zero volume) ellipsoid distance, opposite direction ensure minBack <= maxBack.
priority	**[priority: accessType inputOutput, type SFFloat CDATA "0"]** Browser hint [0..1] to choose which sounds to play.
spatialize	**[spatialize: accessType initializeOnly, type SFBool (true|false) "true"]** Whether to spatialize sound playback relative to viewer. **Hint:** only effective between minimum and maximum ellipsoids.
containerField	**[containerField: NMTOKEN "children"]** 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.

AudioClip node

AudioClip retrieves to an audio file for playing by the parent Sound node

- Sound can also use MovieTexture soundtrack as alternate to AudioClip

AudioClip also provides controls for playback

- *startTime* or *stopTime*, *pauseTime* or *resumeTime*, *isActive*, *isPaused, elapsedTime*, *loop*, *duration*

AudioClip fields 1

description is short text summary of sound clip

loop is boolean whether to play once or repeat

pitch is multiplication factor for playback speed

- Default *pitch*='1', slowdown pitch is smaller than 1, speedup greater than 1, must be greater than zero

url holds one or more equivalent addresses for audio file (or stream) to be retrieved

duration_changed event is sent whenever file is loaded, reporting time needed for full play

- Not affected by *pitch* speedup/slowdown factor

AudioClip fields 2

startTime
stopTime
pauseTime
resumeTime
elapsedTime
isActive
isPaused

These fields are defined the same (and operate the same) as the corresponding fields defined for TimeSensor node in Chapter 7.

Computing current sound time within a source clip:

$$t_{sound} = (now - startTime)\ modulo\ (duration \div pitch)$$

AudioClip fields 3: *url*

Sound source for retrieval defined by *url* field

- Ordered list: one or more addresses
- May be local file, remote file, or streaming source
- Can be monitored by LoadSensor node, e.g.

Browser support for .wav format is required

- MIDI and MP3 support are recommended
- other audio formats are optional
- Can check documentation for browsers of interest
- So far, no streaming protocol required in X3D

SoundLoadSensorTest.x3d - Editor

SoundLoadSensorTest.x3d

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.2//EN" "http://www.web3d.org/specifications/x3d-3.2.dtd">
<X3D profile='Immersive' version='3.2' xmlns:xsd='http://www.w3.org/2001/XMLSchema-instance' xsd:noNamespaceSchema
    <head>
        <meta content='SoundLoadSensorTest.x3d' name='title'/>
        <meta content='UsingLoadSensor to test when AudioClip loading is complete' name='description'/>
        <meta content='19 July 2008' name='created'/>
        <meta content='19 July 2008' name='modified'/>
        <meta content='http://X3dGraphics.com' name='reference'/>
        <meta content='http://www.web3d.org/x3d/content/examples/help.html' name='reference'/>
        <meta content='Copyright 2006, Daly Realism and Don Brutzman' name='rights'/>
        <meta content='X3D book, X3D graphics, X3D-Edit, http://www.x3dGraphics.com' name='subject'/>
        <meta content='http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/SoundLoad
        <meta content='X3D-Edit, https://savage.nps.edu/X3D-Edit' name='generator'/>
        <meta content='../license.html' name='license'/>
    </head>
    <Scene>
        <Sound DEF='SomeSound' maxBack='100' maxFront='100' minBack='10' minFront='10'>
            <AudioClip DEF='WaterSounds' description='Running Water' url='"aqua.wav" "http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-Enviro
        </Sound>
        <LoadSensor DEF='ReportWhenLoaded'>
            <AudioClip USE='WaterSounds' containerField='watchList'/>
        </LoadSensor>
        <ROUTE fromField='loadTime' fromNode='ReportWhenLoaded' toField='startTime' toNode='WaterSounds'/>
        <!-- send completion report to console -->
        <Script DEF='ConsoleReport'>
            <field accessType='inputOnly' name='loadComplete' type='SFBool'/>
            <field accessType='inputOnly' name='loadTime' type='SFTime'/>
<![CDATA[
ecmascript:
function initialize ()
{
        Browser.println ('Script initialize() complete');
}
function loadComplete (value)
{
        Browser.println ('LoadSensor AudioClip isLoaded=' + value);
}
function loadTime (value)
{
        Browser.println ('LoadSensor AudioClip loadTime=' + value);
}
]]>
        </Script>
        <ROUTE fromField='isLoaded' fromNode='ReportWhenLoaded' toField='loadComplete' toNode='ConsoleReport'/>
        <ROUTE fromField='loadTime' fromNode='ReportWhenLoaded' toField='loadTime' toNode='ConsoleReport'/>
        <!-- show visible indication of load waiting, complete -->
```

Edit AudioClip

containerField: source
DEF: WaterSounds
USE:
description: Running Water
loop:
pitch: 1
pauseTime: 0
resumeTime: 0
startTime: 0
stopTime: 0
url: "aqua.wav" "http://X3dGraphics.com/examples/X3dForWebAuthors
OK Cancel Help

Octaga Console

Open on warnings

```
Loading file: 'C:\www.web3d.org\x3d\content\examples\X3dForWebAuthors\Chapter12-EnvironmentSensorSound\SoundLoadSensorTest.x3d'
Script initialize() complete
LoadSensor AudioClip isLoaded=true
LoadSensor AudioClip loadTime=1.21652e+009
```

37:68 INS

■ AudioClip	**AudioClip provides audio data used by <Sound> nodes.** **Hint: add a Sound node first.**
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!
description	**[description: accessType inputOutput, type SFString CDATA #IMPLIED]** text description to be displayed for action of this node. **Hint:** many XML tools substitute XML character references automatically if needed (like & for & or " for ").
url	**[url: accessType inputOutput, type MFString CDATA #IMPLIED]** address, name of sound file. Support for .wav format is required, .midi format is recommended, others are optional. **Hint:** Strings can have multiple values, so separate each string by quote marks. ["http://www.url1.org" "http://www.url2.org" "etc."]. **Hint:** XML encoding for " is " (a character entity). **Warning:** strictly match directory and filename capitalization for http links! **Hint:** can replace embedded blank(s) in url queries with %20 for each blank character.
loop	**[loop: accessType inputOutput, type SFBool (true|false) "false"]** repeat indefinitely when loop=true, repeat only once when loop=false.
pitch	**[pitch: accessType inputOutput, type SFFloat CDATA "1.0"]** Multiplier for the rate at which sampled sound is played. changing pitch also changes playback speed.

startTime	**[startTime: accessType inputOutput, type SFTime CDATA "0"]** Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
stopTime	**[stopTime: accessType inputOutput, type SFTime CDATA "0"]** Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
duration_changed	**[duration_changed: accessType outputOnly, type SFTime CDATA #FIXED ""]** duration_changed is length of time in seconds for one cycle of audio.
isActive	**[isActive: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isActive true/false events are sent when playback starts/stops.
isPaused	**[isPaused: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isPaused true/false events are sent when AudioClip is paused/resumed.
pauseTime	**[pauseTime: accessType inputOutput, type SFTime CDATA "0"]** When time now >= pauseTime, isPaused becomes true and AudioClip becomes paused. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
resumeTime	**[resumeTime: accessType inputOutput, type SFTime CDATA "0"]** When resumeTime becomes <= time now, isPaused becomes false and AudioClip becomes active. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
elapsedTime	**[elapsedTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Current elapsed time since AudioClip activated/running, cumulative in seconds, and not counting any paused time.
containerField	**[containerField: NMTOKEN "source"]** 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.

Chapter Summary

Chapter Summary

Environment Sensors and Sound Nodes

- LoadSensor detects availability of other content
- ProximitySensor detects user location, orientation
- VisibilitySensor detects visibility of region to user
- Sound controls spatialization of audio outputs
- AudioClip controls retrieval and playback of audio files and streams

These sensors can improve animation timeliness

Sound can improve apparent realism of scenes

Suggested exercises

Modify one of your previous animated scenes (that include Inline or ImageTexture), add a LoadSensor to trigger further animation.

Show example use of ProximitySensor and VisibilitySensor nodes as animation triggers.

Add multiple sounds to a scene, locate each with an object, and show sound spatialization while navigating through the scene.

Model a musical instrument so that clicking on the instrument produces tones or music.

Resources and References

Resources 1

Audacity

- Free, open source software for recording and editing sounds. Mac OS X, Windows, GNU/Linux.
- http://audacity.sourceforge.net

MidiEditor

- http://midieditor.sourceforge.net

Midi editor for Netbeans

- http://blogs.oracle.com/geertjan/entry/midi_editor_for_netbeans_ide

Midi Manufacturers Association

- http://www.midi.org (includes a midi tutorial)

Resources 2

Wikipedia

- http://en.wikipedia.org/wiki/Musical_Instrument_Digital_Interface
- http://en.wikipedia.org/wiki/List_of_MIDI_editors_and_sequencers

Freebyte midi file links

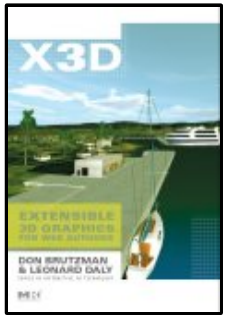
- http://www.freebyte.com/music/#midifiles

Midipedia

- http://en.midipedia.net

References 1

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



- Chapter 12, Environment and Sound Nodes
- http://x3dGraphics.com
- http://x3dgraphics.com/examples/X3dForWebAuthors

X3D Resources

- http://www.web3d.org/x3d/content/examples/X3dResources.html

References 2

X3D-Edit Authoring Tool

- https://savage.nps.edu/X3D-Edit

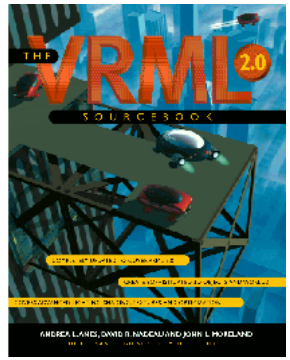
X3D Scene Authoring Hints

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

X3D Graphics Specification

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

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 24 – Sound
- Chapter 27 - Sensing Visibility Proximity Collision

Durand Begault, *3D sound for virtual reality and multimedia*, Academic Press, 1994

- http://portal.acm.org/citation.cfm?id=184407

web3D CONSORTIUM

References 4

Wikipedia: audio

- **Audio codec (coder/decoder)** and **list of codecs**
  http://en.wikipedia.org/wiki/Audio_codec
  http://en.wikipedia.org/wiki/List_of_codecs#Audio_codecs
- **Comparison of audio formats**
  http://en.wikipedia.org/wiki/Comparison_of_audio_formats
- **3D audio effect**
  http://en.wikipedia.org/wiki/3D_audio_effect
- **Digital audio**
  http://en.wikipedia.org/wiki/Digital_audio
- **Surround sound** and **Virtual surround**
  http://en.wikipedia.org/wiki/3D_sound
  http://en.wikipedia.org/wiki/Virtual_surround

Additional references: digital sound

Begault, Durand R., *3D Sound for Virtual Reality and Multimedia*, Morgan Kaufmann Publishers, September 1994.

Collins, Karen, *Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design*, MIT Press, Cambridge Massachusetts, October 2008. http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=11652

Computer Music Journal, MIT Press, Cambridge Massachusetts. http://www.mitpressjournals.org/cmj

Dobrian, Christopher, *Digital Audio*, from MSP: The Documentation, Cycling '74 and IRCAM, December 1997. Available at http://music.arts.uci.edu/dobrian/digitalaudio.htm

International Community for Auditory Display (ICAD). http://icad.org

Storms, Russell, *Auditory-Visual Cross-Modal Perception Phenomena*, Ph.D. Dissertation, Naval Postgraduate School, Monterey California, September 1998. http://bosun.nps.edu/uhtbin/hyperion-image.exe/98Sep_Storms.pdf

Contact

Don Brutzman

brutzman@nps.edu

http://faculty.nps.edu/brutzman

Code USW/Br, Naval Postgraduate School
Monterey California 93943-5000 USA
1.831.656.2149 voice

web|3D CONSORTIUM

CGEMS, SIGGRAPH, Eurographics

The Computer Graphics Educational Materials Source(CGEMS) site is designed for educators

- to provide a source of refereed high-quality content
- as a service to the Computer Graphics community
- freely available, directly prepared for classroom use
- http://cgems.inesc.pt

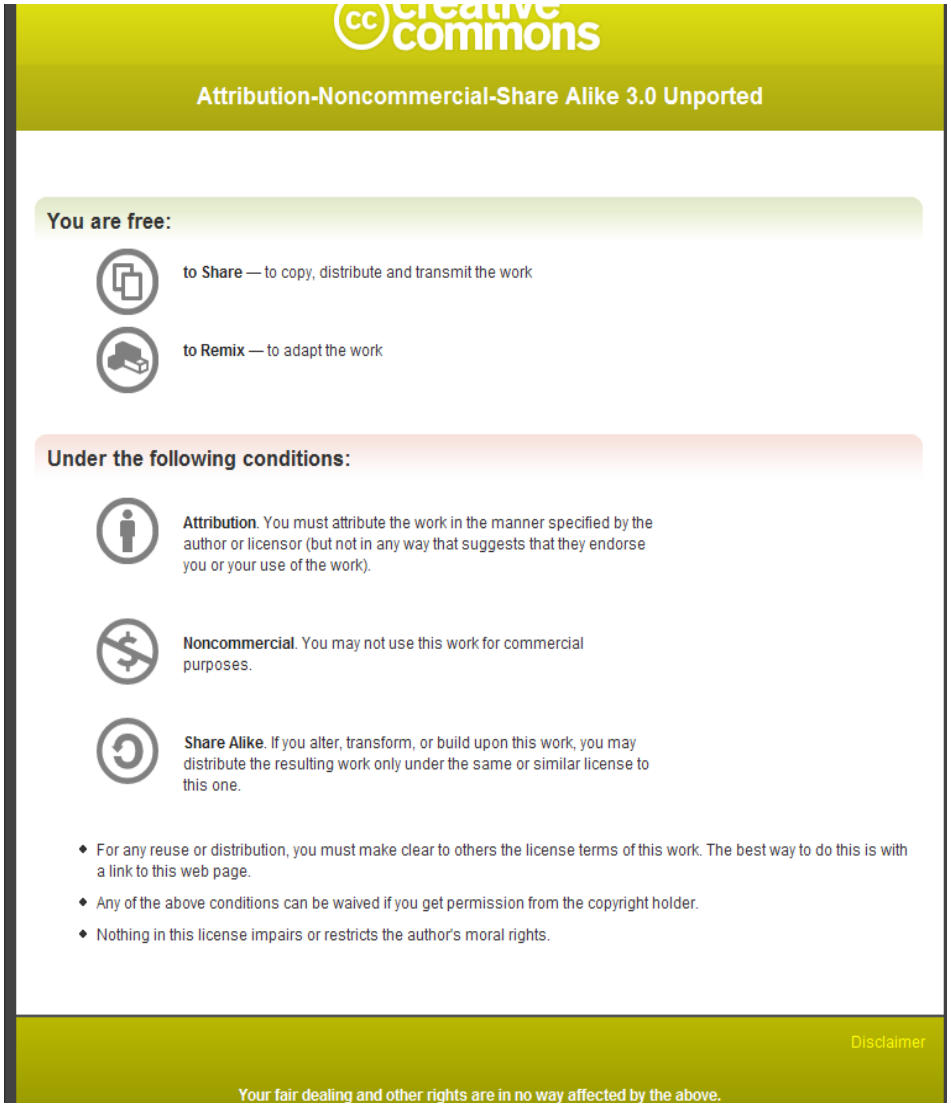
X3D for Web Authors recognized by CGEMS! ☺

- Book materials: X3D-Edit tool, examples, slidesets
- Received jury award for Best Submission 2008

CGEMS supported by SIGGRAPH, Eurographics

Creative Commons open-source license

http://creativecommons.org/licenses/by-nc-sa/3.0



Open-source license
for X3D-Edit software and X3D example scenes

http://www.web3d.org/x3d/content/examples/license.html

X3D Graphics for Web Authors
Chapter 12

Environment Sensor and Sound Nodes

Hereafter, when they come to model heav'n
And calculate the stars, how they will wield
The mighty frame, how build, unbuild, contrive
To save appearances, how gird the sphere
with centric and eccentric scribbled o'er,
Cycle and epicycle, orb in orb.

John Milton, Paradise Lost, 1667

web3D CONSORTIUM

1

Contents

web|3D CONSORTIUM

2

Chapter Overview

3

Overview: Environment Sensor and Sound Nodes

Common fields

- *center*, *size*, *enabled*, *isActive*, *enterTime*, *exitTime*

Nodes

- LoadSensor detects availability of other content
- ProximitySensor detects user location, orientation
- VisibilitySensor detects visibility of region to user
- Sound controls spatialization of audio outputs
- AudioClip controls retrieval and playback of audio files and streams

web|3D CONSORTIUM

4

back to Table of Contents

Concepts

web|3D CONSORTIUM

review

Common field: *enabled*

enabled is an inputOutput boolean field that turns a sensor node on or off

- Thus allowing author to permit or disable flow of user-driven events which drive other responses
- Set *enabled*='false' to disrupt an event chain

Regardless of whether *enabled*='true' a sensor still needs a ROUTE connection from its output, or else no interaction response occurs

web3D CONSORTIUM

Common field: *isActive*

isActive is an outputOnly boolean field that reports when sensor has received user input

- *isActive* true value sent when proximity or visibility conditions are met
- *isActive* false value sent when proximity or visibility conditions are no longer met

Routing *isActive* values can enable, disable TimeSensor and other animation nodes

- Rapid sequencing on/off can be a difficulty, however
- BooleanFilter, BooleanToggle, BooleanTrigger help, described in Chapter 9 Event Utilities and Scripting

Common fields: *center, size*

center and *size* are SFVec3f fields indicating the location and extent of the box that corresponds to the sensed volume of the node

These values are relative to transformation hierarchy created by parent Transform nodes

- *center* can be scaled, translated, rotated
- *size* can be scaled, rotated

This is helpful for DEF/USE in multiple locations

- Warning: no box overlap, or results unpredictable
- *size*='0 0 0' equivalent to *enabled*='false'

web3D CONSORTIUM

8

Common fields: *enterTime*, *exitTime*

enterTime and *exitTime* are SFTime output values sent whenever the sensor proximity or visibility condition is met

- These timestamp values are helpful triggers for other animation chains

Corresponding *isActive* event sent

- true or false, corresponds to *enterTime* or *exitTime*
- accompanying timestamp values also matches

web|3D CONSORTIUM

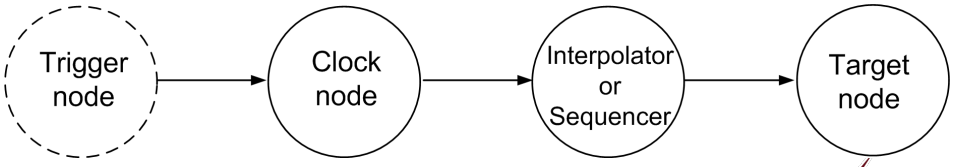
9

The term “accompanying timestamp value” refers to the fact that every event sent includes an embedded time of the the event. These can be captured using a Script node, where the input-event function can have method parameters for both the event value and event timestamp. See Chapter 9 sections on Script nodes for details.

review

Example behavior event chain

- User clicks button to start a timer clock
- Clock outputs new event at start of each frame,
- ... which stimulates linear-interpolation function which produces another output value
- ... which updates some target value in scene graph
- Repeat event traversal after each frame redraw



web3D CONSORTIUM

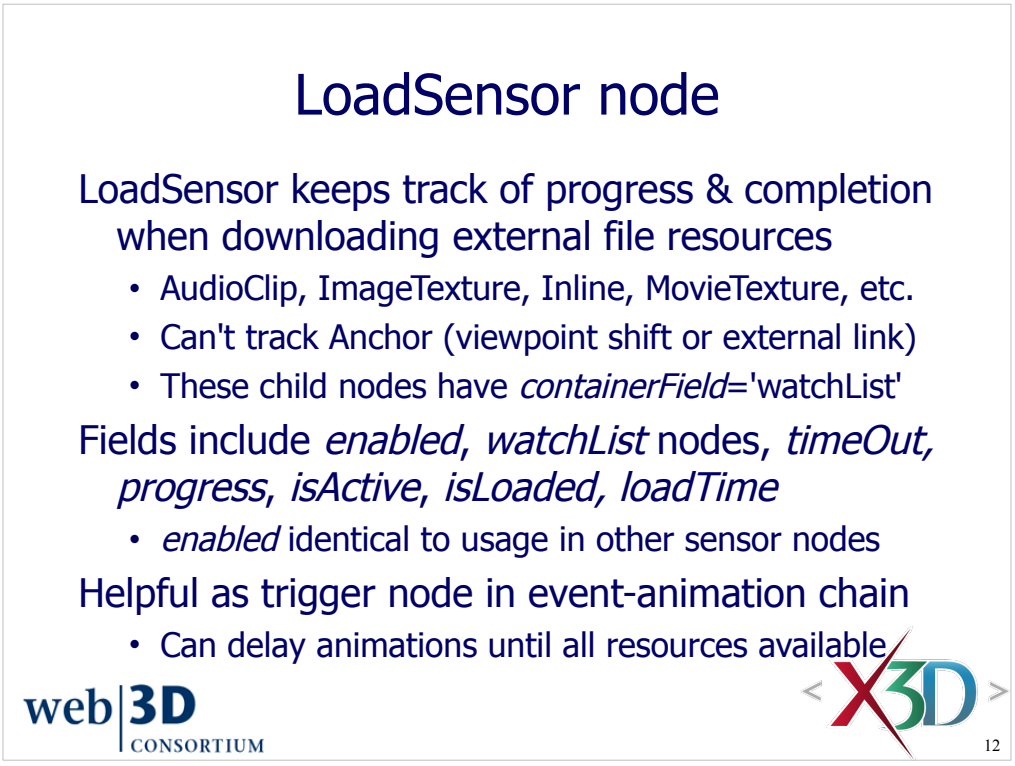
<X3D>

10

back to Table of Contents

X3D Nodes and Examples

11



Full list of nodes with X3DUrlObject that can be tracked by LoadSensor:

- **AudioClip ImageTexture Inline MovieTexture** (all versions of X3D)

plus

- **ImageCubeMapTexture ImageTexture3D PackagedShader ShaderPart** and **ShaderProgram** (v3.1 and later)

LoadSensor fields: *watchList*, *timeOut*

watchList is an MFNode array that lists the nodes of interest to be monitored by LoadSensor

- Usually USE copies of DEF nodes appearing elsewhere in scene
- Each watched node must have *url* field list
- All nodes listed in the *watchList* array must succeed for LoadSensor to succeed

timeOut defines maximum seconds to wait

- default 0 means indefinite, no time limit
- External network source might nevertheless time out if excessive time required

If you want individual control of when nodes load, use multiple LoadSensor nodes.

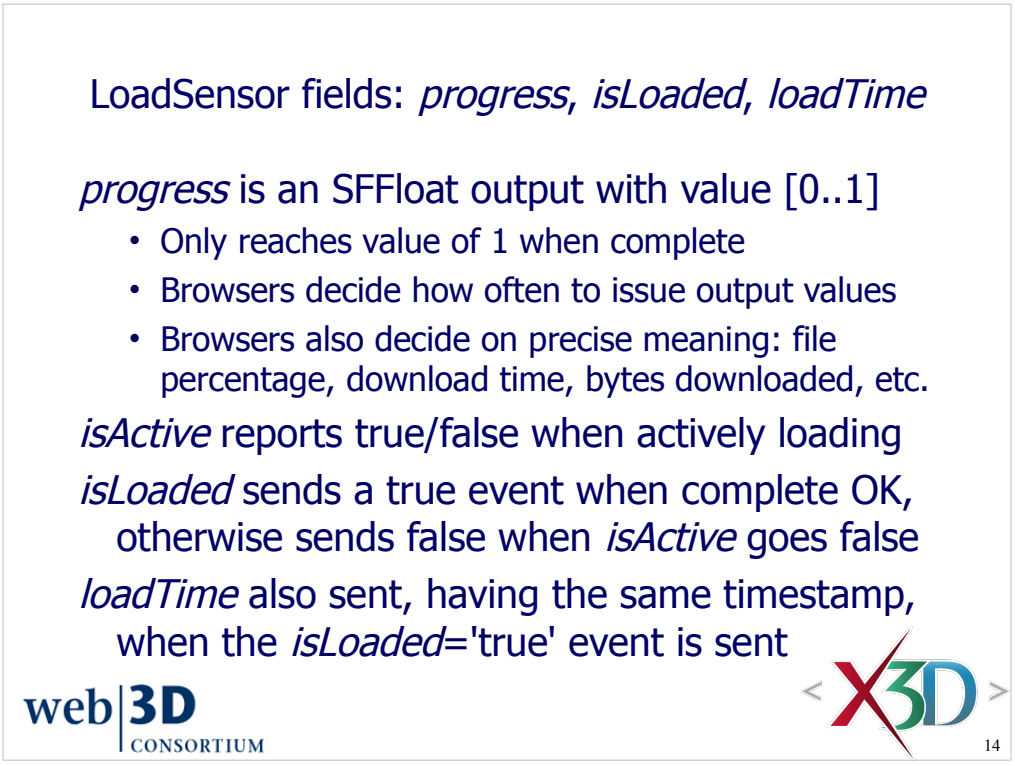
If you want group control of when several nodes load, use a single LoadSensor node.

watchList nodes can be original DEF declarations, but are not rendered. Thus they likely will need a USE copy elsewhere in the scene to accomplish anything.

Each address in *url* list is retrieved sequentially until a successful retrieval occurs.

Note that nodes which are independently defined in the scene are independently downloaded. Thus if two separate ImageTexture nodes refer to a given image, it will be downloaded twice.

It is thus important to DEF/USE copies of nodes that refer to big files, or else download time is twice as long (and memory usage on the user's machine is twice as great). This behavior is required since resources might change over time.



Refer to a given browser's release notes to learn how it handles *progress* measurement.

LoadSensor hints

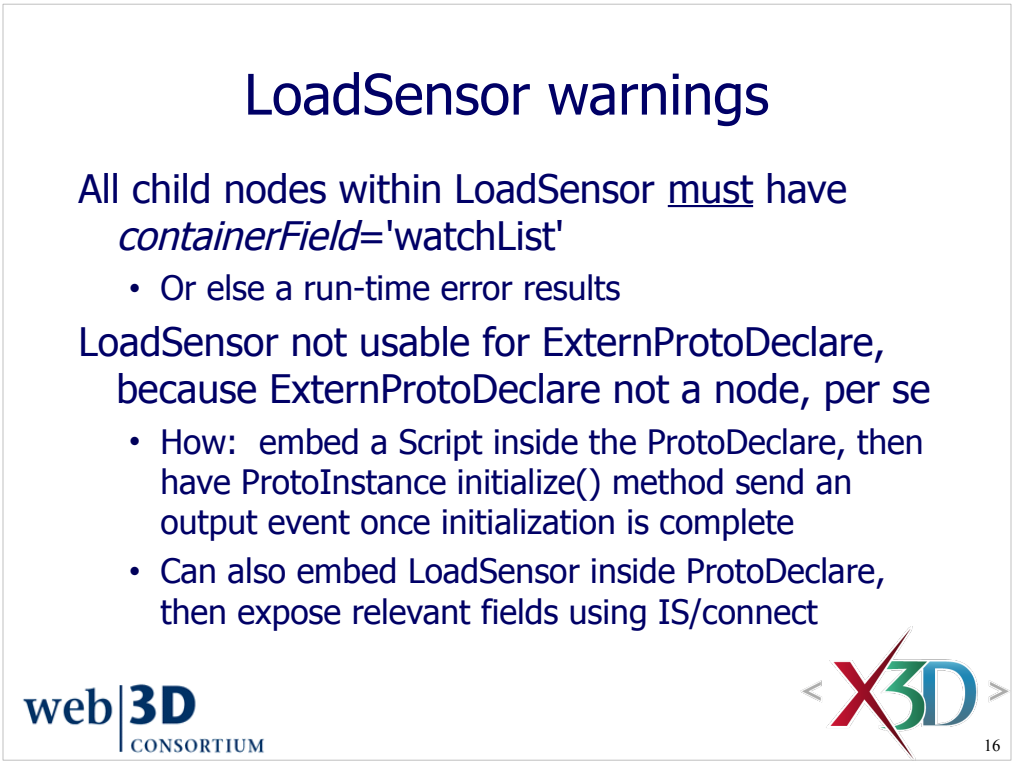
Good design practices:

- Include LoadSensor as trigger in animation chain if target animation makes no sense without resources
- Plan on having a default behavior or warning, just in case the resources don't load

Use multiple LoadSensor nodes for multiple resources, if precise progress results needed

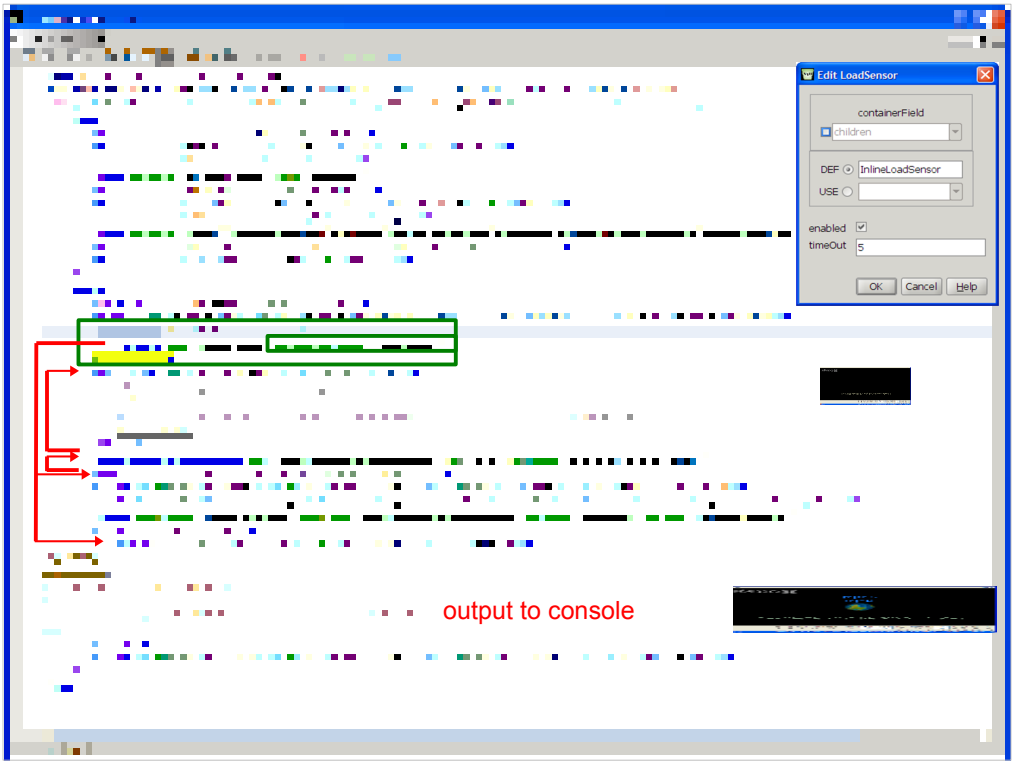
- Can't detect which address in *url* array succeeds
- Can combine with Script to sequence addresses

web|3D CONSORTIUM



LoadSensor functionality can also be included in prototype design by embedding a LoadSensor node, providing it with watchList children that includes the local X3DUrlObject nodes of interest, and then exposing the appropriate LoadSensor fields as ProtoDeclare fields using IS/connect links. Prototypes are described further in Chapter 14, Creating Prototype Nodes.

TODO: Script and prototype examples, plus X3D-Edit interface support



Note that the contained child nodes within LoadSensor (each of which has a URL) must have containerField='watchList' set, rather than the default value of 'children'.

LoadSensor	**LoadSensor generates events as watchList child nodes are either loaded or fail to load. Changing watchlist child nodes restarts the LoadSensor.** **Hint: use multiple LoadSensor nodes to track multiple loading nodes individually.** **Hint: Background is not sensed due to multiple-image ambiguity.** **Warning: watchList child nodes are not rendered, so normally USE copies of other nodes to sense load status.** **Hint: use Inline 'load' field to prompt or defer loading.** **Warning: new X3D node, not supported in VRML 97.**
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!
enabled	**[enabled: accessType inputOutput, type SFBool (true|false) "true"]** Enables/disables node operation.
timeOut	**[timeOut: accessType inputOutput, type SFTime CDATA "0" #IMPLIED]** Time in seconds of maximum load duration prior to declaring failure. Default value zero means use browser defaults.
isActive	**[isActive: outputOnlytype SFBool (true|false) #FIXED ""]** isActive true/false events are sent when load sensing starts/stops.
isLoaded	**[isLoaded: accessType outputOnly, type SFBool (true|false) #FIXED ""]** Notify when all watchList child nodes are loaded, or at least one has failed. Sends true on successfully loading all watchList child nodes. Sends false on timeOut of any watchList child nodes, failure of any watchList child nodes to load, or no local copies available and no network present. Hint: use multiple LoadSensor nodes to track multiple loading nodes individually.
loadTime	**[loadTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time of successful load complete, not sent on failure.
progress	**[progress: accessType outputOnly, type SFFloat CDATA [0.0 .. 1.0]** #FIXED ""] Sends 0.0 on start and 1.0 on completion. Intermediate values are browser dependent and always increasing (may indicate fraction of bytes, fraction of expected time or another metric). Hint: only 0 and 1 events are guaranteed.
containerField	**[containerField: NMTOKEN "children"]** 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.

ProximitySensor node

ProximitySensor tracks user position, orientation within box defined by author as active volume

- Tracking box is not visibly rendered
- Tracking box is affected by parent transformations

Can track (and react to) user navigation in scene

- Able to send multiple events: *position_changed*, *orientation_changed*, *centerOfRotation_changed*

Fields also include *center*, *size*, *enabled*, *isActive*, *enterTime*, *exitTime*

Helpful as trigger node in event-animation chain

- Can delay animations until user is close to action

Output events

position_changed reports location relative to ProximitySensor *center*

- Best is to keep *center* at origin, with very large *size*

orientation_changed events sent when user's view rotates within the monitored volume

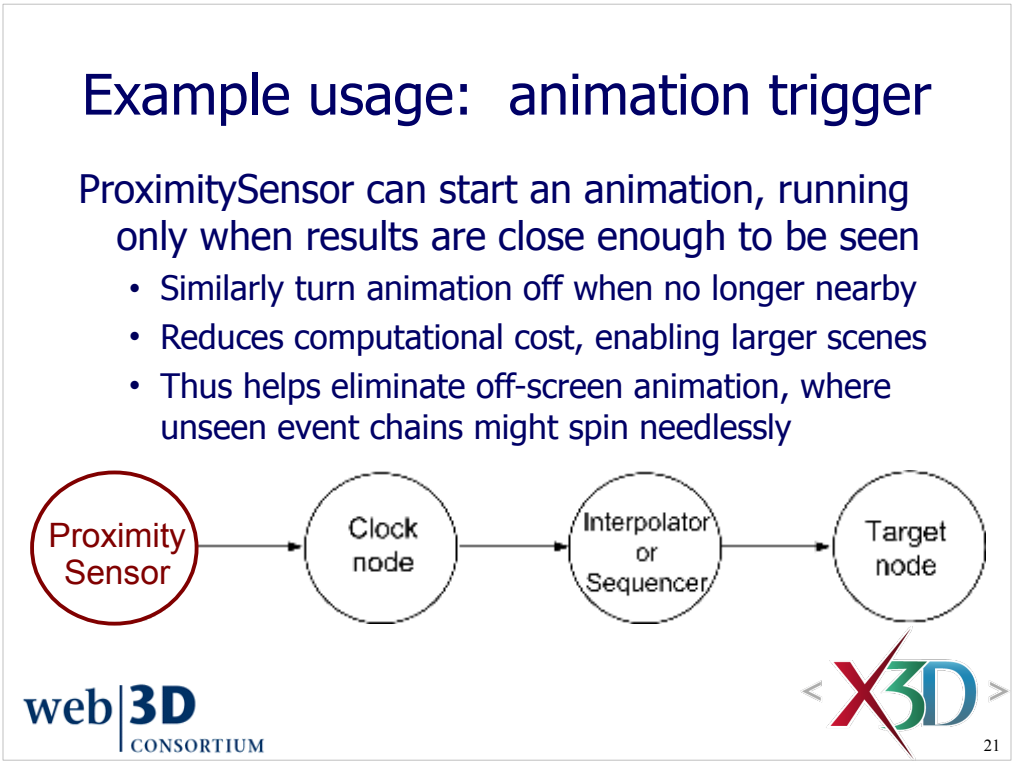
- Also occurs if proximity box rotates independently

centerOfRotation_changed monitors output of NavigationInfo if LOOKAT point changes

- Viewer must be in LOOKAT mode for this to occur

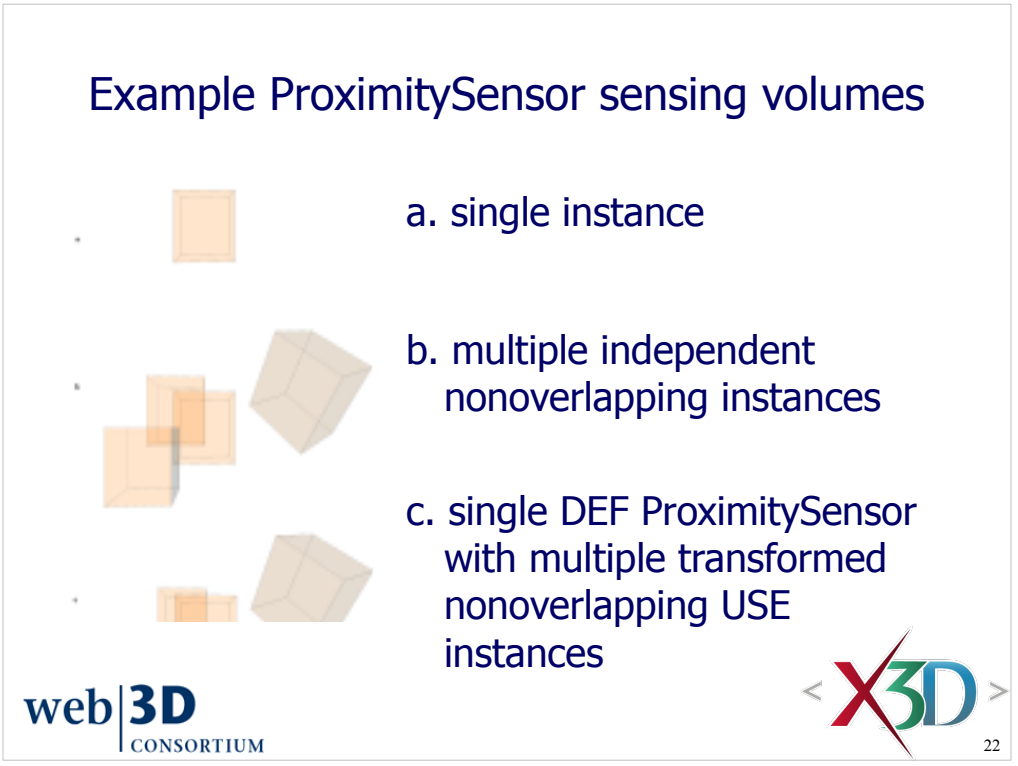
web|3D CONSORTIUM

20



In this case the trigger node is a VisibilitySensor.

- can ROUTE isActive to TimeSensor.enabled, or else
- can ROUTE enterTime to TimeSensor.startTime and also ROUTE exitTime to TimeSensor.stopTime

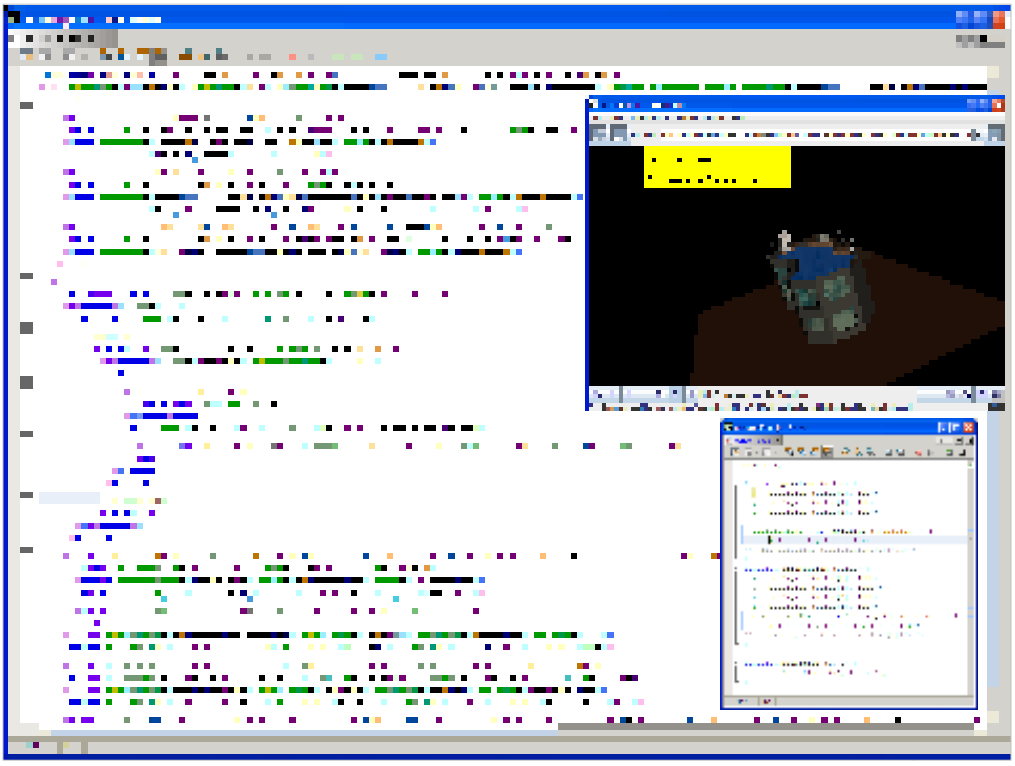


X3D for Web Authors, Figure 12.1, page 337.

Three example scenes:

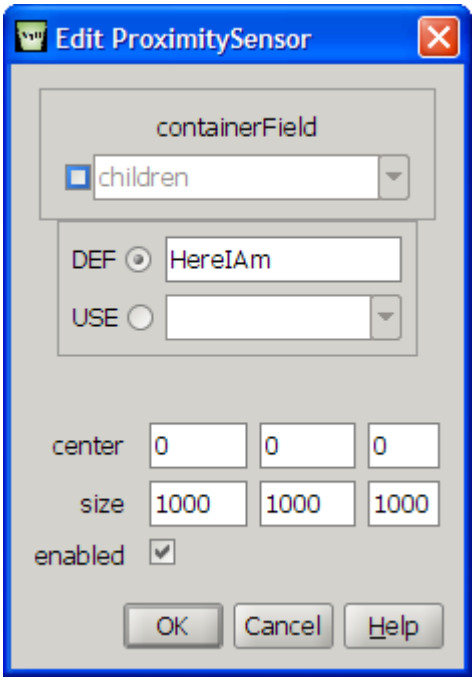
(a) ProximitySensorSingle.x3d
(b) ProximitySensorMultiple.x3d
(c) ProximitySensorNoOverlap.x3d

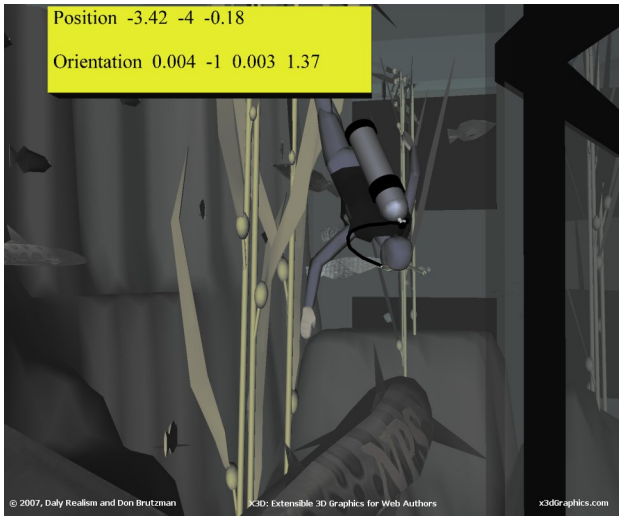
Available in chapter directory
http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/ProximitySensor.x3d

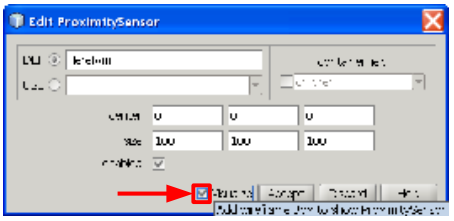
Figure 12.2, p. 338, ProximitySensor reporting position and orientation in a Heads-Up Display (HUD), stabilizing the HUD screen location.

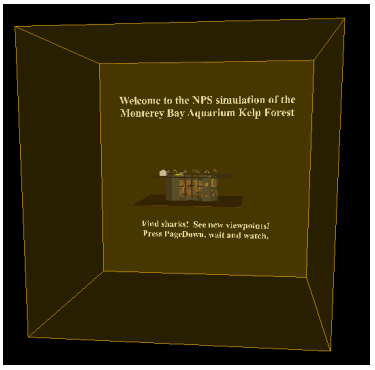




ProximitySensor visualization

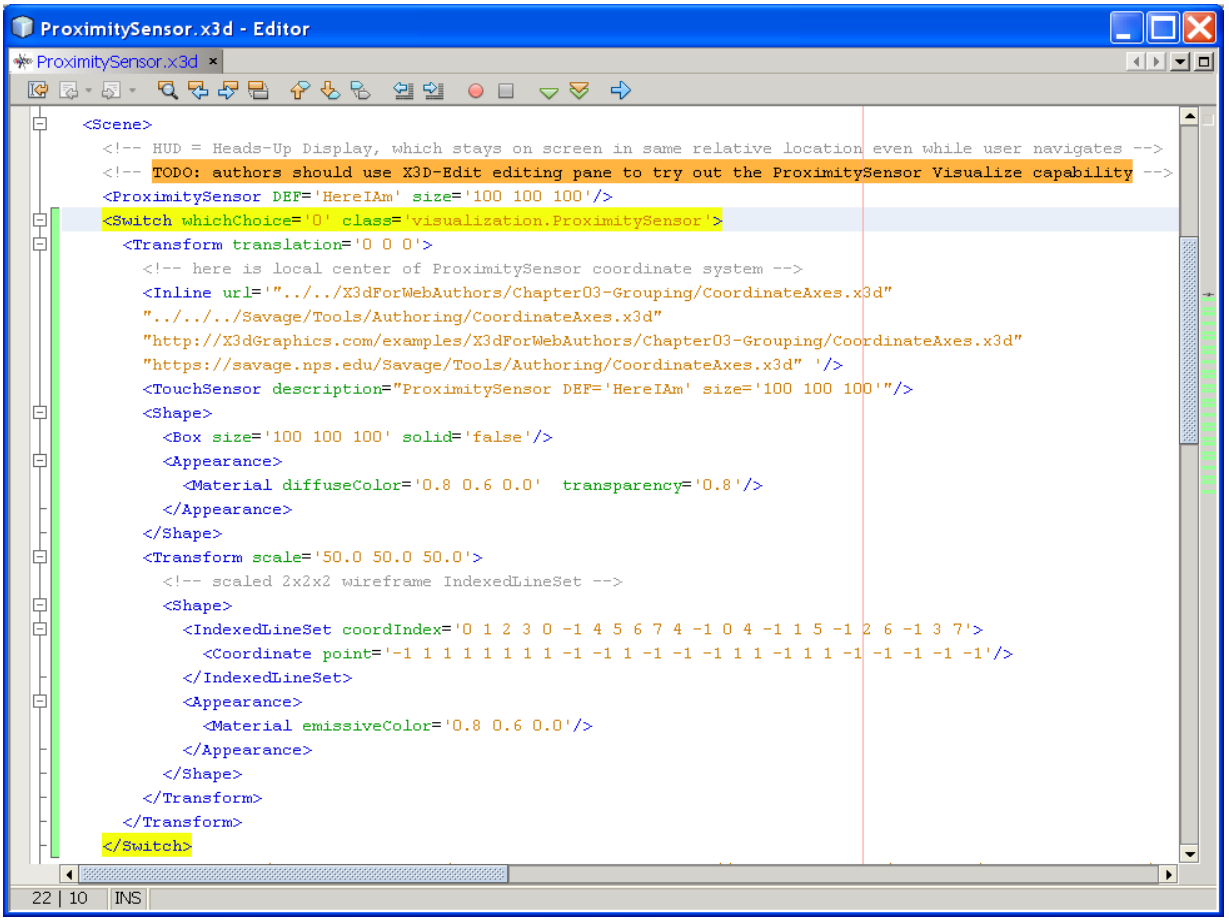
- Visualization option provided by X3D-Edit
  - Displays transparent wireframe box outlining ProximitySensor *size* boundaries
  - Also displays coordinate axes at *center* location
  - `<ProximitySensor DEF='HereIAm' size='100 100 100'/>`





ProximitySensor visualization box inserted into example scene

http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/ProximitySensor.x3d



ProximitySensor	**ProximitySensor generates events when the viewer enters, exits and moves within a region of space (defined by a box).** **Hint: multiple USEd instances are cumulative, but avoid overlaps.** **Hint: can first use Transform to relocate/reorient box.** **Hint: surround entire world to start behaviors once scene is loaded.**
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!
enabled	**[enabled: accessType inputOutput, type SFBool (true|false) "true"]** Enables/disables node operation.
center	**[center: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** Position offset from origin of local coordinate system.
size	**[size: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** size of Proximity box. Hint: size 0 0 0 is same as enabled false.
isActive	**[isActive: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isActive true/false events are sent as viewer enters/exits Proximity box. isActive=true when viewer enters Proximity box, isActive=false when viewer exits Proximity box.
position_changed	**[position_changed: accessType outputOnly, type SFVec3f CDATA #FIXED ""]** Sends translation event relative to center.
orientation_changed	**[orientation_changed: accessType outputOnly, type SFRotation CDATA #FIXED ""]** Sends rotation event relative to center.
centerOfRotation_changed	**[centerOfRotation_changed: accessType outputOnly, type SFRotation CDATA #FIXED ""]** Sends changed centerOfRotation values, likely caused by user interaction.
enterTime	**[enterTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera enters the box.
exitTime	**[exitTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera exits the box.
containerField	**[containerField: NMTOKEN "children"]** 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.

VisibilitySensor node

VisibilitySensor detects whether a given volume of space is visible from current user view

- Depends on user's current direction and position
- Not dependent on proximity distance or range rate
- Intermediate occluding geometry has no effect

Fields include *center*, *size*, *enabled*, *isActive*, *enterTime*, *exitTime*

Helpful as trigger node in event-animation chain

- Can delay animations until user is able and ready to view them

web3D CONSORTIUM

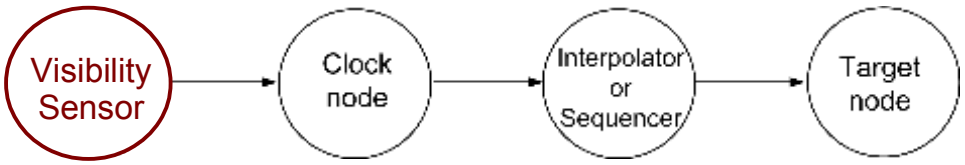
26

Intermediate occluding geometry means any blocking shapes that are between the user's camera and the VisibilitySensor node center. Thus any geometry in the scene (whether moving or stable) has no effect on VisibilitySensor activation.

Example usage: animation trigger

VisibilitySensor can control animation, only running when results are visible

- Similarly turn animation off when no longer visible
- Reduces computational cost, enabling larger scenes



web3D CONSORTIUM

27

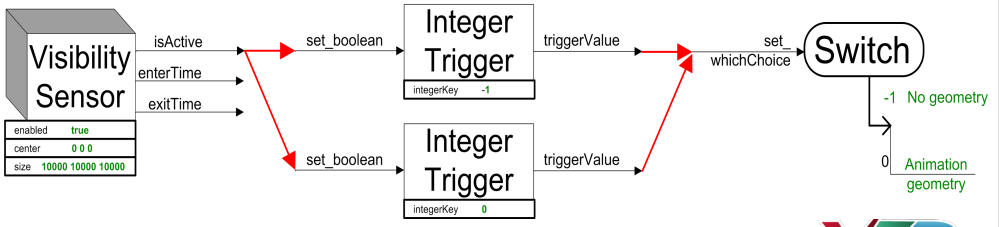
In this case the trigger node is a VisibilitySensor.

- can ROUTE isActive to TimeSensor.enabled, or else
- can ROUTE enterTime to TimeSensor.startTime and also ROUTE exitTime to TimeSensor.stopTime

Example usage: switch out geometry

VisibilitySensor can remove scene subgraphs, only showing them when results are visible

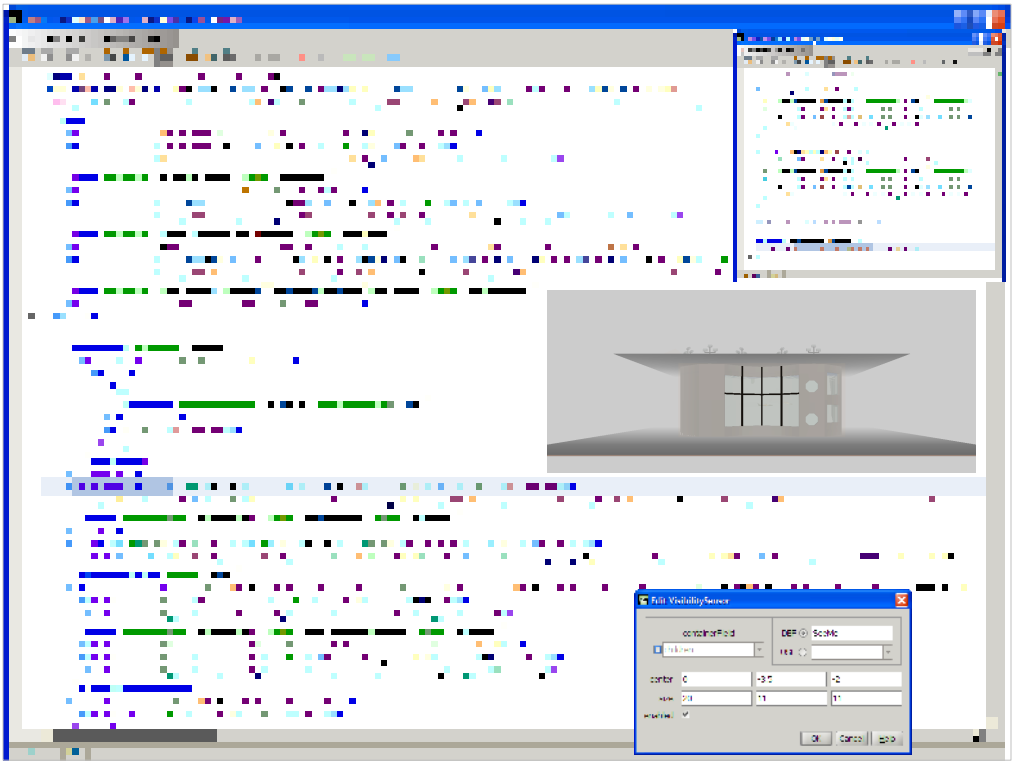
- Similarly removing them when no longer visible
- Reduces computational cost, enabling larger scenes



web3D CONSORTIUM

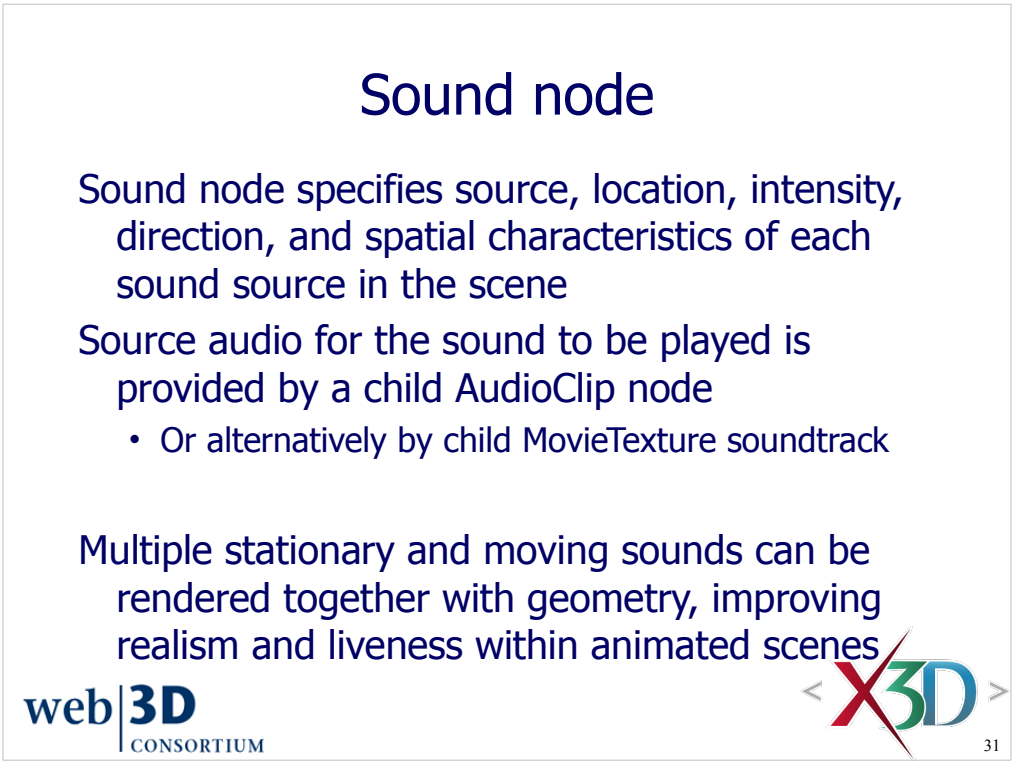
28

Red arrows in the above diagram are ROUTE connections



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/VisibilitySensor-KelpForestMain.x3d

VisibilitySensor	**VisibilitySensor detects when user can see a specific object or region as they navigate the world. This region is bounded by a box.** **Hint: often used to attract user attention or improve performance.** **Hint: Sensors are affected by peer nodes and children of peers.**
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!
enabled	**[enabled: accessType inputOutput, type SFBool (true|false) "true"]** Enables/disables node operation.
center	**[center: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** Translation offset from origin of local coordinate system.
size	**[size: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** size of visibility box, measured from center in meters.
isActive	**[isActive: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isActive true/false events are sent when triggering the sensor. isActive=true when entering visibility region, isActive=false when exiting visibility region.
enterTime	**[enterTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera enters visibility region for sensor.
exitTime	**[exitTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Time event generated when user's camera exits visibility region for sensor.
containerField	**[containerField: NMTOKEN "children"]** 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.



The Sound and AudioClip nodes make an integral pair and work closely together.

AudioClip is a separate node in order to decouple file and stream loading from the spatial characteristics of sound in the scene.

Sound fields: *location, direction, intensity, priority*

location is center position of sound origin

- Relative to local coordinate system

direction is unit-vector direction of sound axis

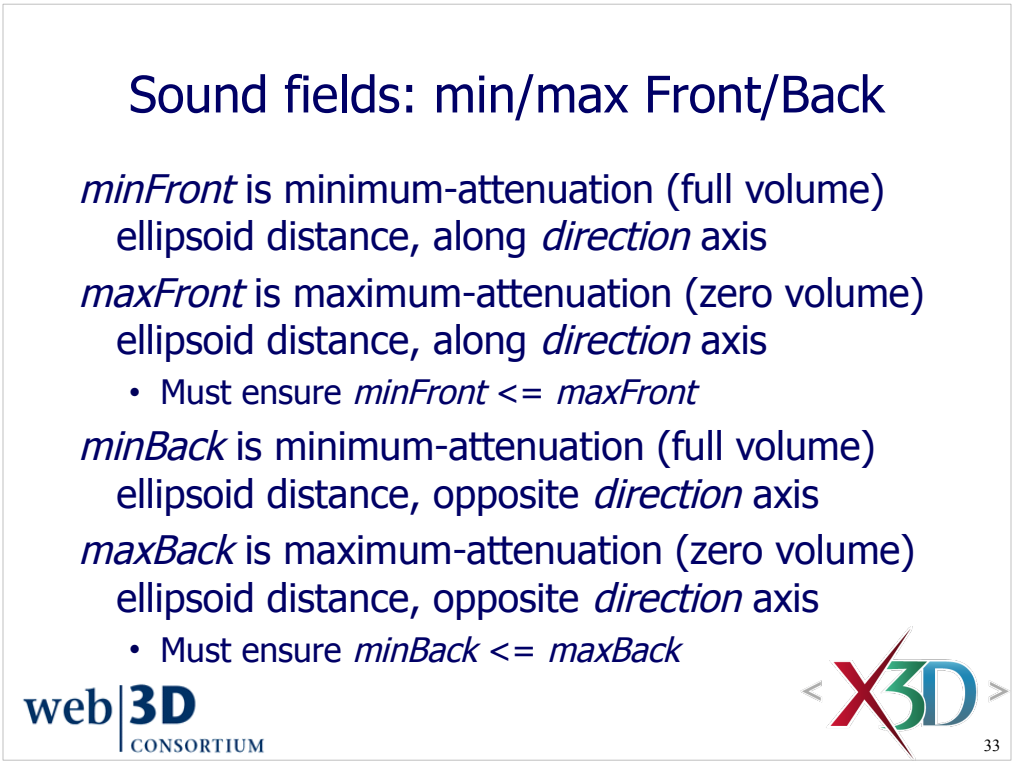
- Three-tuple vector, not a four-tuple SFRotation
- Relative to local coordinate system

intensity is factor [0..1] adjusts loudness of emitted sound

- Multiplied against original volume level of source

priority [0..1] is a browser hint, if ever needed, to choose which of several sounds to play

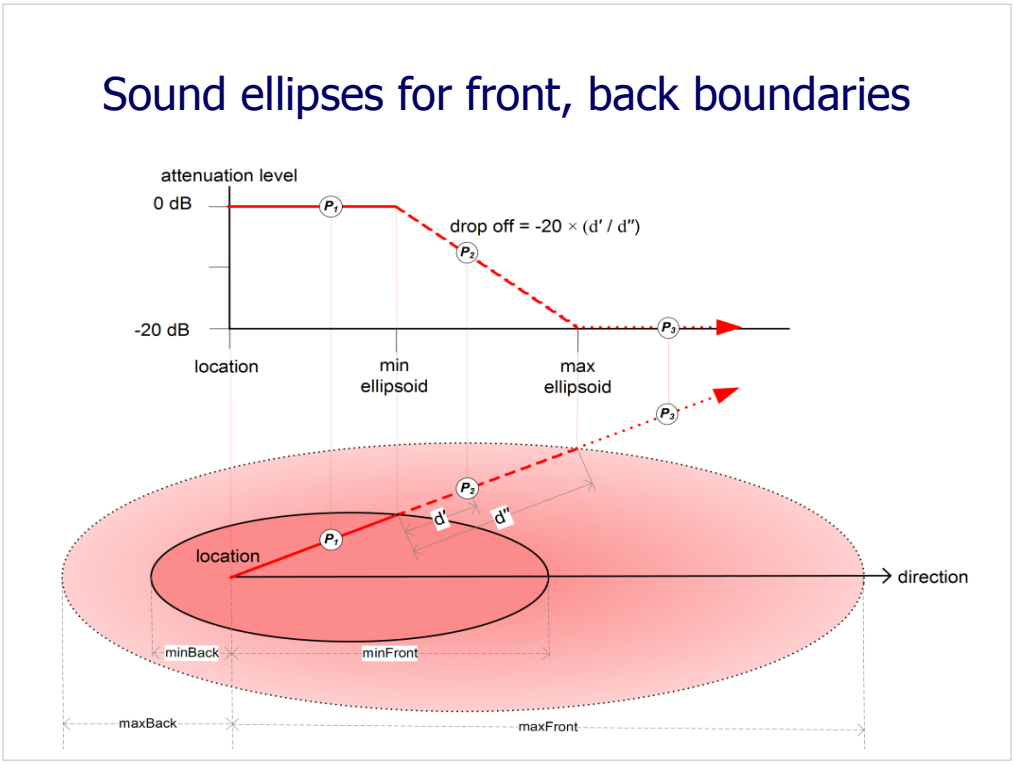
web3D CONSORTIUM



These field names can be confusing.

Hints:

- Minimum refers to inner ellipsoid (full audio volume)
- Maximum refers to outer ellipsoid (zero audio volume)
- Front means along the direction vector
- Back means opposite to the direction vector



p. 343, Figure 12.4. Sound ellipsoids correspond to linear spatialization boundaries for attenuation of Sound node intensity.

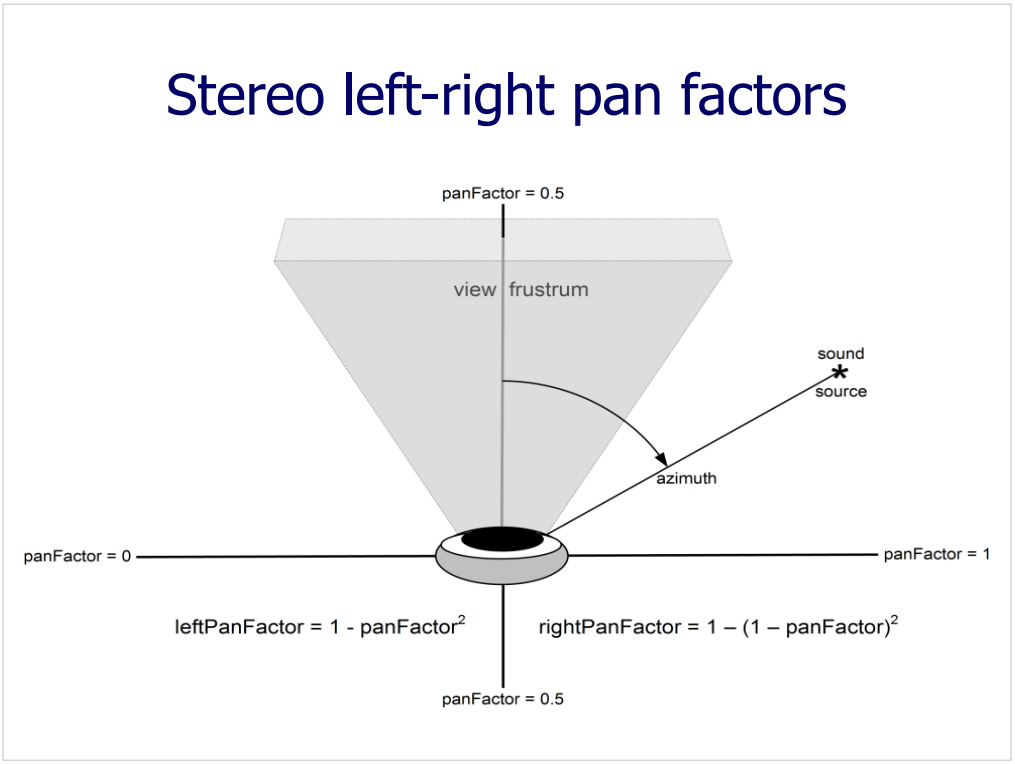
Sound field: *spatialize*

spatialize is a boolean determining whether to spatialize sound playback relative to viewer

- Only acts between minimum, maximum ellipsoids
- Stereo effect

Pan-factor computations determine spatialization effect on each channel

web|3D CONSORTIUM



p. 342, Figure 12.3. Stereo-panning algorithm for attenuation of sound intensity is based on the azimuth angle relative to the user's current view direction.

Computing ellipsoid dimensions

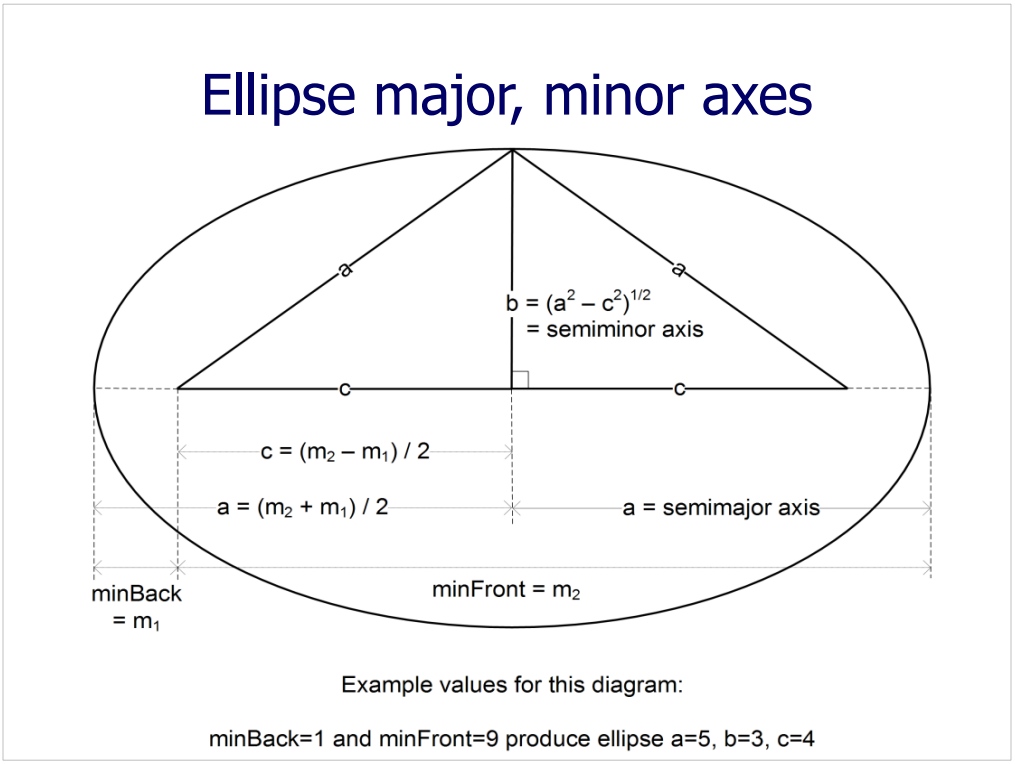
location is focus point of the sound ellipsoid

maxFront, *maxBack* define dimensions of outer ellipsoid along and opposite the *direction* axis

Note however that ellipsoid width is not defined by these parameters, because that value is dependent on the other definitions

- The next diagram, equations show how to calculate

web|3D CONSORTIUM



p. 344, Figure 12.5. Derivation and example values for an ellipse semi-minor axis, given focus location and front/back distances. The upper plot shows intensity values corresponding to boundaries in the lower diagram.

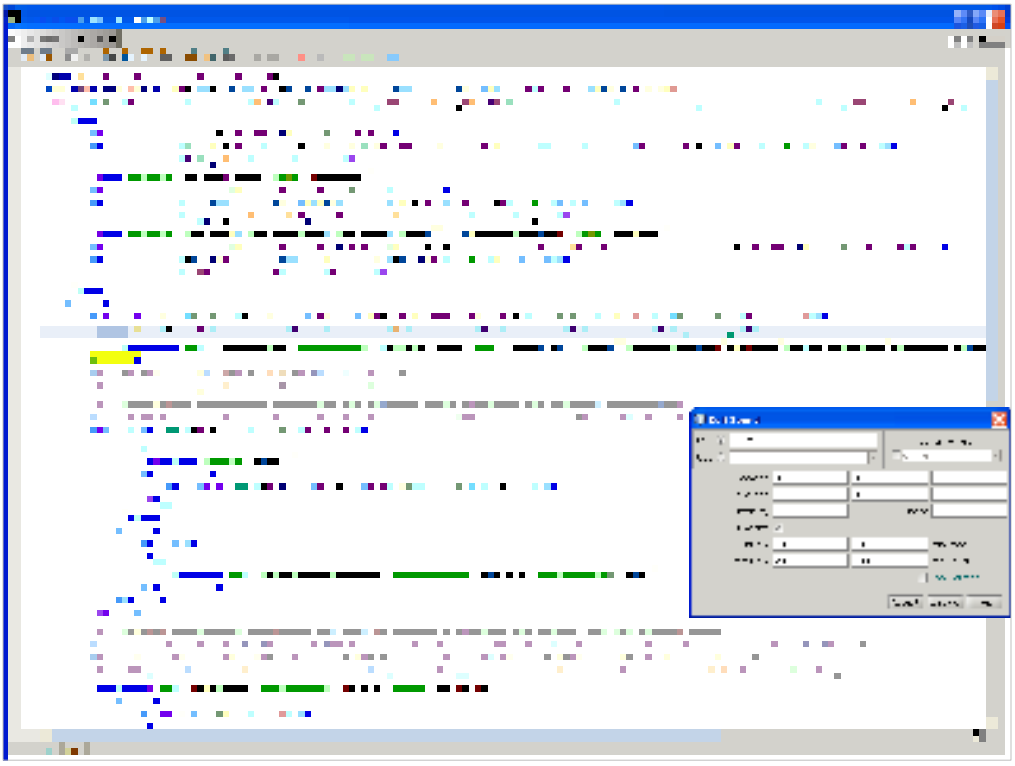
Calculating the width of sound ellipsoid

$$
\begin{aligned}
minHalfWidth = b_{\min} &= \sqrt{a^2 - c^2} \\
&= \sqrt{\left[\frac{m_1 + m_2}{2}\right]^2 - \left[\frac{m_1 - m_2}{2}\right]^2} \\
&= \frac{1}{2}\sqrt{m_1^2 + 2m_1m_2 + m_2^2 - \left(m_1^2 - 2m_1m_2 + m_2^2\right)} \\
&= \frac{1}{2}\sqrt{2m_1m_2 + 2m_1m_2} \\
&= \frac{1}{2}(2)\sqrt{m_1m_2} = \sqrt{m_1m_2} \\
minHalfWidth &= \sqrt{minBack \cdot minFront}
\end{aligned}
$$

similarly

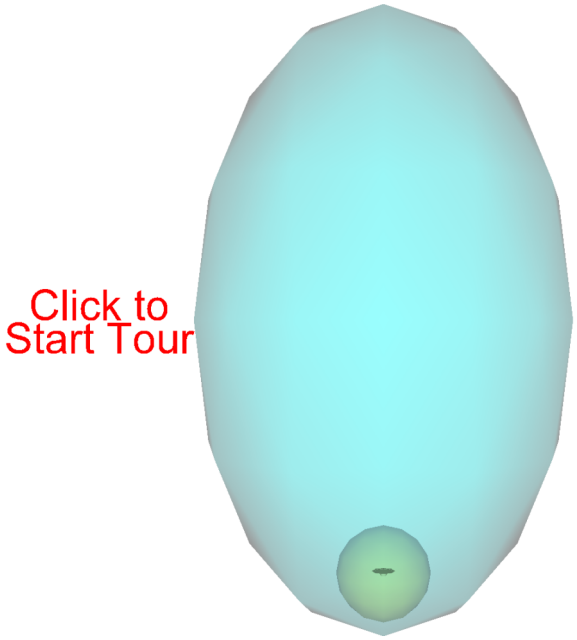
$$maxHalfWidth = \sqrt{maxBack \cdot maxFront}$$

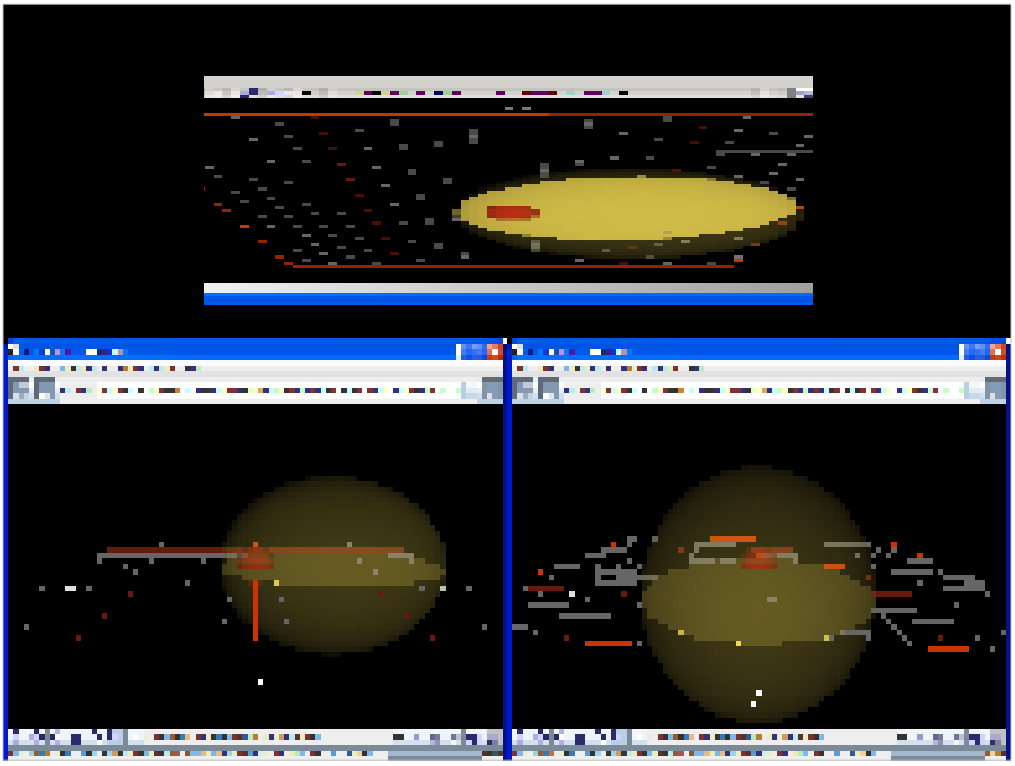
p. 345, Figure 12.6. Derivation of ellipsoid minHalfWidth and maxHalfWidth.



http://X3dGraphics.com/examples/X3dForWebAuthors/Chapter12-EnvironmentSensorSound/SoundAudioClip.x3d

This is an improved example for Figure 12.7 (shown below).





Three views of this example. Navigate within the scene to hear variations from minimum attenuation (full volume) within the red sphere, then reducing down to maximum attenuation (zero volume) once at our outside the yellow spheroid.

Note that the non-uniformly scaled sphere is an approximation for the sound ellipse. Also note that Sound nodes do not have a visible manifestation for their geometry.

This visualization example was constructed manually. The following visualization example was constructed automatically using the author-assist feature for Sound that is provided by X3D-Edit.

Sound node hints

Since ellipses shrink quickly, raise up the Sound node's location to match Viewpoint eye level

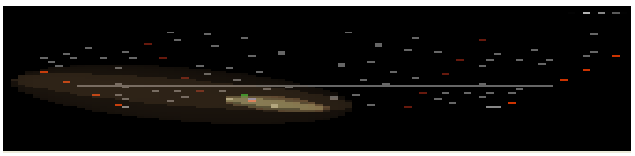
- When navigating in WALK mode, a typical *avatarSize* height is 1.6m
- Example: `<Sound location='0 1.6 0'/>`

Large volumes are good if you want to ensure the user hears what is intended

X3D-Edit includes a visualization author assist to transparently renders minimum (full volume) and maximum (zero audio volume) ellipsoids

web3D CONSORTIUM

Sound node visualization



```
<Sound location='-3 1.6 -3' direction='1 0 1' minBack='5' maxBack='6' minFront='1' maxFront='10'/>
```

X3D-Edit author-assist feature for Sound node automatically adds ellipsoid visualization to a scene

web3D CONSORTIUM

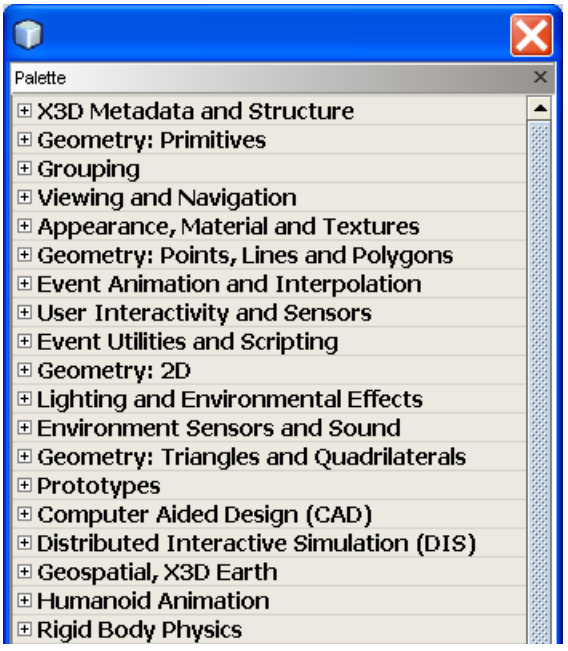
<X3D>

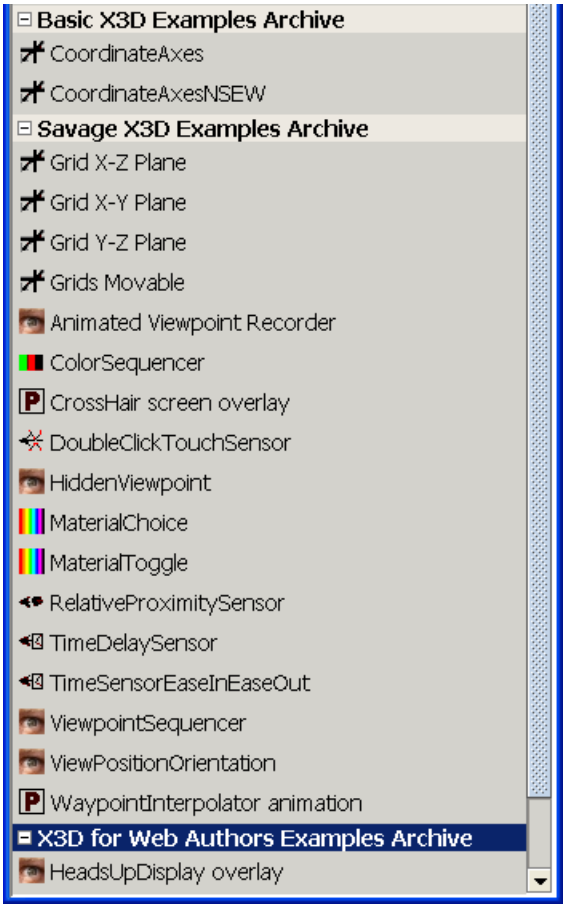
43

Notice the following visualization properties for the Sound ellipsoid:

- Proper location and direction
- The inner minimum (full volume) ellipsoid tends to the left of the local origin along the back direction, while
- The outer maximum (zero audio volume) ellipsoid tends to the right of the local origin along the front direction

Various coordinate axes grids are easily available via the X3D-Edit palettes for the Basic and Savage X3D Archive models.





Sound	Sound contains an AudioClip or MovieTexture for sound playback. 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!
location	**[location: accessType inputOutput, type SFVec3f CDATA "0 0 0"]** Position of sound center, relative to local coordinate system.
direction	**[direction: accessType inputOutput, type SFVec3f CDATA "0 0 1"]** direction of sound axis, relative to local coordinate system.
intensity	**[intensity: accessType inputOutput, type SFFloat CDATA "1"]** Factor [0..1] adjusting loudness (decibels) of emitted sound.
minFront	**[minFront: accessType inputOutput, type SFFloat CDATA "1"]** Minimum-attenuation (full volume) ellipsoid distance, along direction ensure minFront <= maxFront.
minBack	**[minBack: accessType inputOutput, type SFFloat CDATA "1"]** Minimum-attenuation (full volume) ellipsoid distance, opposite direction ensure minBack <= maxBack.
maxFront	**[maxFront: accessType inputOutput, type SFFloat CDATA "10"]** Maximum-attenuation (zero volume) ellipsoid distance, along direction ensure minFront <= maxFront.
maxBack	**[maxBack: accessType inputOutput, type SFFloat CDATA "10"]** Maximum-attenuation (zero volume) ellipsoid distance, opposite direction ensure minBack <= maxBack.
priority	**[priority: accessType inputOutput, type SFFloat CDATA "0"]** Browser hint [0..1] to choose which sounds to play.
spatialize	**[spatialize: accessType initializeOnly, type SFBool (true|false) "true"]** Whether to spatialize sound playback relative to viewer. **Hint:** only effective between minimum and maximum ellipsoids.
containerField	**[containerField: NMTOKEN "children"]** 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.

AudioClip node

AudioClip retrieves to an audio file for playing by the parent Sound node

- Sound can also use MovieTexture soundtrack as alternate to AudioClip

AudioClip also provides controls for playback

- *startTime* or *stopTime*, *pauseTime* or *resumeTime*, *isActive*, *isPaused, elapsedTime*, *loop*, *duration*

web|3D CONSORTIUM

AudioClip fields 1

description is short text summary of sound clip

loop is boolean whether to play once or repeat

pitch is multiplication factor for playback speed

- Default *pitch*='1', slowdown pitch is smaller than 1, speedup greater than 1, must be greater than zero

url holds one or more equivalent addresses for audio file (or stream) to be retrieved

duration_changed event is sent whenever file is loaded, reporting time needed for full play

- Not affected by *pitch* speedup/slowdown factor

web3D CONSORTIUM

47

duration_changed='-1' means that audio data is not yet loaded or not available

AudioClip fields 2

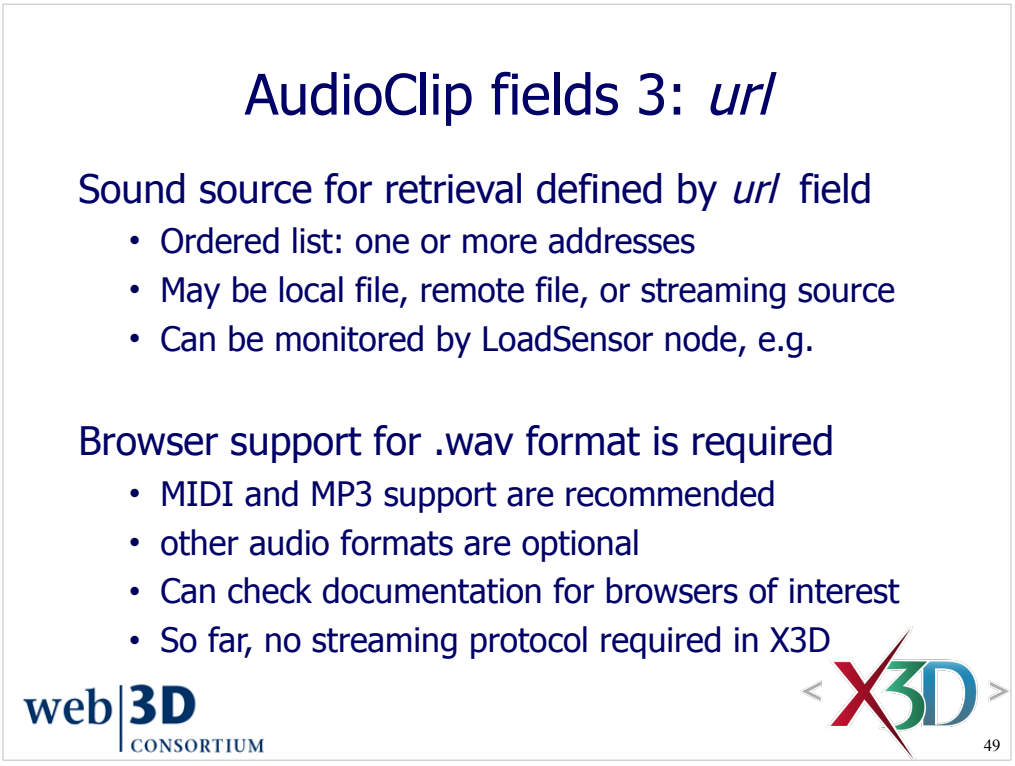
startTime
stopTime
pauseTime
resumeTime
elapsedTime
isActive
isPaused

These fields are defined the same (and operate the same) as the corresponding fields defined for TimeSensor node in Chapter 7.

Computing current sound time within a source clip:

$$t_{sound} = (now - startTime)\ modulo\ (duration \div pitch)$$

web|3D CONSORTIUM



X3D Specification reference:

- X3D Abstract Specification, Sound component, section 16.4.1 AudioClip

http://www.web3d.org/x3d/specifications/ISO-IEC-FDIS-19775-1.2-X3D-AbstractSpecification/Part01/components/sound.html#AudioClip

WAV format reference:

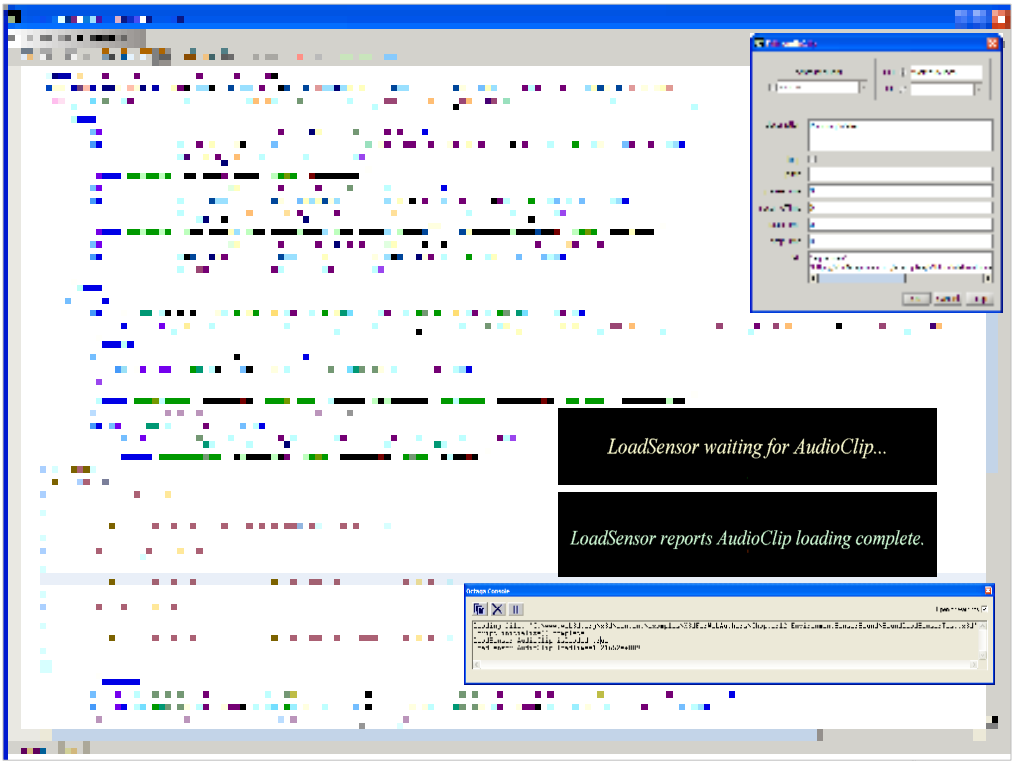
- Waveform Audio File Format, Multimedia Programming Interface and Data Specification v1.0, Issued by IBM & Microsoft, 1991. ftp://ftp.cwi.nl/pub/audio/RIFF-format
- TODO: this does not seem authoritative, need to revise reference

MIDI Specification reference:

- Complete MIDI 1.0 Detailed Specification v96.1 (second edition), MIDI Manufacturers Association, P.O. Box 3173, La Habra, CA 90632-3173 USA, 2001. http://www.midi.org

MP3 Specification reference:

- ISO/IEC 11172-1:1993, Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/seconds - Part 1: Systems.
- TODO: need online reference


LoadSensor waiting for AudioClip...
LoadSensor reports AudioClip loading complete.

AudioClip	AudioClip provides audio data used by <Sound> nodes. Hint: add a Sound node first.
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!
description	[description: accessType inputOutput, type SFString CDATA #IMPLIED] text description to be displayed for action of this node. Hint: many XML tools substitute XML character references automatically if needed (like & for & or " for ").
url	[url: accessType inputOutput, type MFString CDATA #IMPLIED] address, name of sound file. Support for .wav format is required, .midi format is recommended, others are optional. Hint: Strings can have multiple values, so separate each string by quote marks. ["http://www.url1.org" "http://www.url2.org" "etc."]. Hint: XML encoding for " is " (a character entity). Warning: strictly match directory and filename capitalization for http links! Hint: can replace embedded blank(s) in url queries with %20 for each blank character.
loop	[loop: accessType inputOutput, type SFBool (true|false) "false"] repeat indefinitely when loop=true, repeat only once when loop=false.
pitch	[pitch: accessType inputOutput, type SFFloat CDATA "1.0"] Multiplier for the rate at which sampled sound is played. changing pitch also changes playback speed.

startTime	**[startTime: accessType inputOutput, type SFTime CDATA "0"]** Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
stopTime	**[stopTime: accessType inputOutput, type SFTime CDATA "0"]** Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
duration_changed	**[duration_changed: accessType outputOnly, type SFTime CDATA #FIXED ""]** duration_changed is length of time in seconds for one cycle of audio.
isActive	**[isActive: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isActive true/false events are sent when playback starts/stops.
isPaused	**[isPaused: accessType outputOnly, type SFBool (true|false) #FIXED ""]** isPaused true/false events are sent when AudioClip is paused/resumed.
pauseTime	**[pauseTime: accessType inputOutput, type SFTime CDATA "0"]** When time now >= pauseTime, isPaused becomes true and AudioClip becomes paused. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
resumeTime	**[resumeTime: accessType inputOutput, type SFTime CDATA "0"]** When resumeTime becomes <= time now, isPaused becomes false and AudioClip becomes active. Absolute time: number of seconds since Jan 1, 1970, 00:00:00 GMT. **Hint:** usually receives a ROUTEd time value.
elapsedTime	**[elapsedTime: accessType outputOnly, type SFTime CDATA #FIXED ""]** Current elapsed time since AudioClip activated/running, cumulative in seconds, and not counting any paused time.
containerField	**[containerField: NMTOKEN "source"]** 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.

back to Table of Contents

Chapter Summary

Chapter Summary

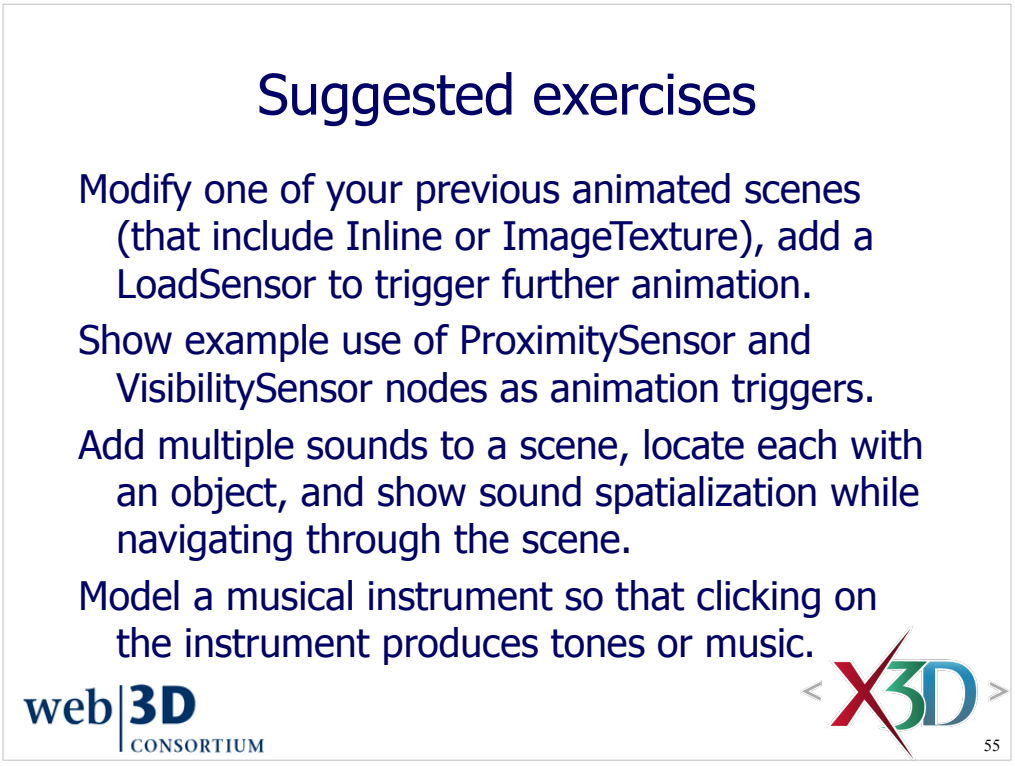
Environment Sensors and Sound Nodes

- LoadSensor detects availability of other content
- ProximitySensor detects user location, orientation
- VisibilitySensor detects visibility of region to user
- Sound controls spatialization of audio outputs
- AudioClip controls retrieval and playback of audio files and streams

These sensors can improve animation timeliness

Sound can improve apparent realism of scenes

web3D CONSORTIUM



Example ProximitySensor and Visibility scenes can be found in the *VRML 2.0 Sourcebook:*
http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook/Chapter27-SensingVisibilityProximityCollision

Figure 27.1 Visibility Sensor Dungeon Sliding Doors

Figure 27.2 Proximity Sensor Dungeon Sliding Doors

Further example Sound and AudioClip scenes are also found in the *VRML 2.0 Sourcebook:*
http://www.web3d.org/x3d/content/examples/Vrml2.0Sourcebook/Chapter24-Sound

Figure 24.2 Ambient Sound Emitter Markers

Figure 24.3 Touch Sensor Triggered Sound

Figure 24.4 Four Key Keyboard

Figure 24.5 Two Ambient Circling Sounds

Figure 24.6 Directed Ambient Sound

Figure 24.7 Virtual TV

Here is an interesting student example:
http://www.web3d.org/x3d/content/examples/Basic/StudentProjects/KeyboardEightyEightKeys.x3d

Eight-eight key keyboard, extrapolated from VRML Sourcebook Figure 24-4, including animation of key movements coordinated with sounding of key when touched.

back to Table of Contents

Resources and References

Resources 1

Audacity

- Free, open source software for recording and editing sounds. Mac OS X, Windows, GNU/Linux.
- http://audacity.sourceforge.net

MidiEditor

- http://midieditor.sourceforge.net

Midi editor for Netbeans

- http://blogs.oracle.com/geertjan/entry/midi_editor_for_netbeans_ide

Midi Manufacturers Association

- http://www.midi.org (includes a midi tutorial)

Resources 2

Wikipedia

- http://en.wikipedia.org/wiki/Musical_Instrument_Digital_Interface
- http://en.wikipedia.org/wiki/List_of_MIDI_editors_and_sequencers

Freebyte midi file links

- http://www.freebyte.com/music/#midifiles

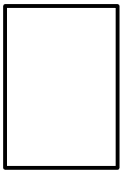
Midipedia

- http://en.midipedia.net

web|3D CONSORTIUM

References 1

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



- Chapter 12, Environment and Sound Nodes
- http://x3dGraphics.com
- http://x3dgraphics.com/examples/X3dForWebAuthors

X3D Resources

- http://www.web3d.org/x3d/content/examples/X3dResources.html

web|3D CONSORTIUM

References 2

X3D-Edit Authoring Tool

- https://savage.nps.edu/X3D-Edit

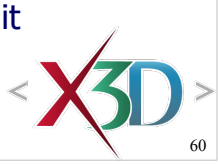
X3D Scene Authoring Hints

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

X3D Graphics Specification

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

web|3D CONSORTIUM



60

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 24 – Sound
- Chapter 27 - Sensing Visibility Proximity Collision

Durand Begault, *3D sound for virtual reality and multimedia*, Academic Press, 1994

- http://portal.acm.org/citation.cfm?id=184407

web3D CONSORTIUM

61

References 4

Wikipedia: audio

- **Audio codec (coder/decoder)** and **list of codecs**
  http://en.wikipedia.org/wiki/Audio_codec
  http://en.wikipedia.org/wiki/List_of_codecs#Audio_codecs
- **Comparison of audio formats**
  http://en.wikipedia.org/wiki/Comparison_of_audio_formats
- **3D audio effect**
  http://en.wikipedia.org/wiki/3D_audio_effect
- **Digital audio**
  http://en.wikipedia.org/wiki/Digital_audio
- **Surround sound** and **Virtual surround**
  http://en.wikipedia.org/wiki/3D_sound
  http://en.wikipedia.org/wiki/Virtual_surround

Additional references: digital sound

Begault, Durand R., *3D Sound for Virtual Reality and Multimedia*, Morgan Kaufmann Publishers, September 1994.

Collins, Karen, *Game Sound: An Introduction to the History, Theory, and Practice of Video Game Music and Sound Design*, MIT Press, Cambridge Massachusetts, October 2008. http://mitpress.mit.edu/catalog/item/default.asp?ttype=2&tid=11652

Computer Music Journal, MIT Press, Cambridge Massachusetts. http://www.mitpressjournals.org/cmj

Dobrian, Christopher, *Digital Audio*, from MSP: The Documentation, Cycling '74 and IRCAM, December 1997. Available at http://music.arts.uci.edu/dobrian/digitalaudio.htm

International Community for Auditory Display (ICAD). http://icad.org

Storms, Russell, *Auditory-Visual Cross-Modal Perception Phenomena*, Ph.D. Dissertation, Naval Postgraduate School, Monterey California, September 1998. http://bosun.nps.edu/uhtbin/hyperion-image.exe/98Sep_Storms.pdf

Contact

Don Brutzman

brutzman@nps.edu

http://faculty.nps.edu/brutzman

Code USW/Br, Naval Postgraduate School
Monterey California 93943-5000 USA
1.831.656.2149 voice

web|3D CONSORTIUM

64

CGEMS, SIGGRAPH, Eurographics

The Computer Graphics Educational Materials Source(CGEMS) site is designed for educators

- to provide a source of refereed high-quality content
- as a service to the Computer Graphics community
- freely available, directly prepared for classroom use
- http://cgems.inesc.pt

X3D for Web Authors recognized by CGEMS! ☺

- Book materials: X3D-Edit tool, examples, slidesets
- Received jury award for Best Submission 2008

CGEMS supported by SIGGRAPH, Eurographics


From the CGEMS home page:

- http://cgems.inesc.pt

  Welcome to CGEMS - Computer Graphics Educational Materials Source. The CGEMS site is designed for educators to provide a source of refereed high-quality content as a service to the Computer Graphics community as a whole. Materials herein are freely available and directly prepared for your classroom.

List of all published modules:

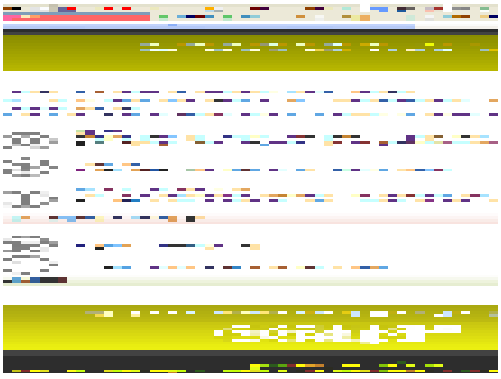
- http://cgems.inesc.pt/authors/ListModules.aspx

CGEMS Editorial Policy:

- http://cgems.inesc.pt/EditorialPolicy.htm

Creative Commons open-source license

http://creativecommons.org/licenses/by-nc-sa/3.0



Open-source license
for X3D-Edit software and X3D example scenes

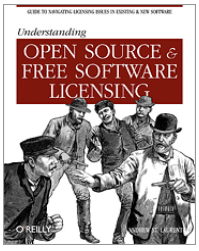
http://www.web3d.org/x3d/content/examples/license.html

License available at

http://www.web3d.org/x3d/content/examples/license.txt

http://www.web3d.org/x3d/content/examples/license.html

Good references on open source:



Andrew M. St. Laurent, *Understanding Open Source and Free Software Licensing*, O'Reilly Publishing, Sebastopol California, August 2004. http://oreilly.com/catalog/9780596005818/index.html

Herz, J. C., Mark Lucas, John Scott, *Open Technology Development: Roadmap Plan*, Deputy Under Secretary of Defense for Advanced Systems and Concepts, Washington DC, April 2006. http://handle.dtic.mil/100.2/ADA450769