### Maya/VS/Edge宿主的“面向3D的.x3d文件宿客”（字符语法主导）

表 6‑24 Maya/VS/Edge宿主的“面向3D的.x3d文件宿客”（字符语法主导）

|  |  |  |
| --- | --- | --- |
| ASP.Net MVC架构的视图V的创新研发：“客户端的Edge浏览器宿主的Blazor宿主的视图V宿客”的C#语言思维  （本表聚焦Maya/VS可以编辑的HTML可以引用的.x3d文件） | ASP.Net MVC架构的控制C的创新研发：“互动的Kestrel/IIS宿主的控制C宿客”的C#语言思维 | ASP.Net MVC架构的模型M的创新研发：“服务端的SS/ChatGPT/Qwen宿主的模型M宿客”的C#语言思维 |
| 基于XML规范的多媒体应用有HTML、SVG/XAML、X3D等，可形成媒体的整体解决方案。本节将系统叙述XML三维图形动画应用--- X3D的基本原理，为三维图形动画的具体应用奠定原理基础。建议结合5.2节所述，进行上机实践探究如下。   1. 理解X3D是XML三维图形动画媒体应用。 2. 掌握X3D常用元素的功能和用法。 3. 掌握三维图形的布局机制。 4. 掌握X3D形体原理，特别是NURBS标准。 5. 掌握X3D形体的外貌原理，材料、贴图。 6. 掌握X3D光照原理。 7. 掌握X3D视点原理。 8. 理解X3D动画原理。 9. 理解X3D外部媒体引用。 10. 掌握X3D超链接。 11. 了解X3D DOM，X3D结合Jscript、C#等编程。 | | |

1. **X3D与Contact平台：基于XML标准的3D应用**

三维图形及其动画是三维形象化信息传播的媒体，由于人类生活的真实世界是三维的，所以三维图形动画是最接近和最能满足真实世界信息传播需要的媒体，代表了媒体的最高境界和媒体技术的发展方向。三维图形动画编辑理论及其实现软件正日益发展并日趋标准化，XML在三维图形动画的具体应用--X3D语言则是描述三维图形动画的Web标准，当前最高版本3.0（http://www.web3d.org/x3d/specifications/x3d/）。（Contact是呈现 X3D 的浏览器（但一般以插件的形式由主浏览器调用运行）

X3D 是eXtensibel 3 Dimention的缩写，中文意译为可扩展3D，是一个描述Web三维图形动画的语言。X3D三维图形动画技术拥有当前流行的三维图形动画的基本功能，还具有可交换性、交互性、可扩展性、易维护性等优点，可完美结合HTML、脚本语言等实现强大的Web多媒体应用，奠定了现代媒体技术领域中三维图形动画技术的基础。目前文本数据媒体技术领域以HTML为技术平台，X3D则注重于结合HTML解决三维应用的问题（参见**Error! Reference source not found.**节）。

X3D 和XAML在语法、功能等众多方面都具有相似性，只不过XAML偏重于二维图形理念，而X3D偏重于三维图形理念，将X3D与XAML对比理解，是掌握X3D的捷径。

X3D文档属于互动式三维图形动画视觉媒体，基本原理可参见**Error! Reference source not found.**中与互动式三维图形动画紧密相关的内容，特别是三维布局、时间特性、交互性等。该部分则将以代码的方式叙述X3D，X3D功能强大、元素及其特性丰富，应该注重掌握X3D的基本原理，并结合Maya软件，进行实践锻炼。

小练习：X3D是VRML（Virtul Reality Modeling Language）的基于XML的发展成果，VRML诞生于XML之先，所以并不完全遵守XML规范，通过网络探究VRML、X3D的发展历史。

1. **X3D文档浏览**

X3D文档的浏览和XML文档的浏览类似（参见**Error! Reference source not found.**）。不过IE6.0不能直接浏览X3D文档，必须安装X3D浏览器（如Bitmanagement公司的BS Contact）作为插件，IE（主浏览器）调用X3D插件浏览、解析、显示X3D文档。由于Web上一般以HTML文档为平台，所以实际上，X3D文档往往作为一个外部媒体被HTML文档引用（参见四、），IE浏览HTML文档时将调用X3D浏览器浏览被引用的X3D文档，**Error! Reference source not found.**节所述的图 6.3‑80就是X3D浏览的一个示例，如果右击鼠标，将弹出右键菜单，分别选择右键菜单“移动|行走”、“移动|检视”、“移动|平移”等，然后上下左右拖动鼠标，滚动鼠标，可体验身临其境的感觉。

X3D文档包含形体、环境、灯光、视点、功能等信息。和HTML、XAML类似，X3D能够把各种媒体有机结合在一起，并具有超链接，交互性等动态功能。但更高境界的是X3D是真实世界的虚拟现实模拟，使用者能够体会到三维空间、灯光、视觉、立体声、距离感等身临其境的功能。因为是模拟真实世界的三维图形，用户自己是一个融合在三维图形世界之中的的观察者（事实上虚拟三维世界中有一个“化身(avatar，谐音：阿凡达)”代表用户），用户可在图形世界之中任意漫游并从任何方向角度观察该方体，体会身临其境的感觉。

IE调用BS Contact插件可全屏或非全屏浏览X3D文档，可参见**Error! Reference source not found.**节。

1. **X3D文档编辑**

X3D文档本质上是文本文档，所以原则上可以使用任何文本编辑器编辑（如Windows自带的记事本程序），但为了提高编辑效率，一般使用原生X3D文档编辑软件进行编辑（如美国Pinecoast公司的Swirl3D），或者使用综合图形编辑软件进行编辑并输出为X3D格式（如Alias公司的Maya，将在**Error! Reference source not found.**详述）。

编辑X3D文档时，通过X3D元素生成三维形体、三维文本，然后对形体进行布局、附予外貌而获得形体的视觉结果；赋予背景、光照、视点获得真实世界感受；通过引用外部媒体而将点阵图、声音、视频与矢量图结合为一个有机的整体；随时间呈现不同的三维对象，将三维对象的属性随时间变化而获得动画效果；结合编程语言获得交互性；从而生成强大的X3D三维图形动画应用。

1. **X3D文档基本结构**

X3D本质上是一种基于XML规范的描述性语言，该语言完全遵守XML语法，可编写并形成X3D三维图形动画文档。一个X3D文档往往对应一个计算机文件，文件扩展名是" .x3d" ，压缩文件的扩展名是".x3dz"。

下面是一个简单X3D文档的示例，反映了X3D文档的基本结构(参见示例文件BasicStructure.x3d)。在浏览器中的运行运行结果将如图 6.3‑80所示。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?>**<!--该代码行声明本文档是一个XML文档，文档编码是utf-8。-->** 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd">**<!--该代码行声明文档架构-->** 4. <X3D version="3.0">**<!--该代码行声明该文档是一个X3D文档，X3D的版本是3.0。-->** 5. <Scene>**<!--<Scene>元素开始是X3D文档主体，<Scene>元素可以嵌套X3D元素而获得强大的三维图形动画应用-->** 6. <Transform> 7. <Shape> 8. <Text string="多媒体技术"/>**<!--生成文本-->** 9. </Shape> 10. <Shape> 11. <Sphere />**<!--生成球体-->** 12. <Appearance> 13. <ImageTexture url="wwwEdu.jpg" />**<!--引用同路径下的JPG图像文件贴图-->** 14. </Appearance> 15. </Shape> 16. </Transform> 17. </Scene> 18. </X3D> |



图 6.3‑80 BasicStructure.x3d在浏览器中的运行运行结果

1. **X3D文档架构及其元素概述**

X3D是一个Web标准，X3D具有标准的预定义的架构（而XML是自定义架构），掌握X3D的关键就是掌握X3D架构中预定义的元素及其特性的功能和用法，X3D架构中预定义的元素及其特性丰富，功能强大，在此主要掌握其基本原理，然后可基于基本原理，参考X3D架构标准（http://www.web3d.org/x3d/specifications/x3d/），自主上机实验，全面掌握X3D。

为了更好地理解X3D素，在此将X3D所有常用元素归类概述如下，以作大概了解。实际应用时，应该根据正编辑的X3D文档的需求而考虑元素及其属性选用，如果不知道元素的具体用法，则应该参考X3D标准。如果以软件为辅助工具，则更可大大地提高效率，使用Maya软件编辑X3D将在**Error! Reference source not found.**详述。

* 1. 布局（Layout）变换类元素

Transform

布局变换类元素为X3D生成布局系统及其变换，典型地如平移、放缩、旋转等，将在0节使用到Transform元素布局变换几何体。

* 1. 几何体（Geometry）类元素

Arc2D、ArcClose2D、Box、Circle2D、Cone、Cylinder、Disk2D、ElevationGrid、Extrusion、GeoElevationGrid、PointSet、IndexedLineSet、IndexedFaceSet、IndexedTriangleFanSet、IndexedTriangleSet、IndexedTriangleStripSet、LineSet、NurbsCurve、NurbsPatchSurface、NurbsSweptSurface、NurbsSwungSurface 、NurbsTrimmedSurface 、Polyline2D、Polypoint2D、Rectangle2D、Sphere、Text 、TriangleFanSet、TriangleSet、TriangleSet2D、TriangleStripSet。

几何体类元素用来生成几何体。如**Error! Reference source not found.**节将使用到PointSet、IndexedLineSet、IndexedFaceSet构建正四面体；三、节将使用到Box、Cone、Cylinder构建立体体等基本几何体；八、节将使用到Text 构建三维文本。

* 1. 几何体（Geometry）类元素的子元素

Color、ColorRGBA、Coordinate、CoordinateDouble、FontStyle、GeoCoordinate、MultiTextureCoordinate 、Normal、TextureCoordinate、TextureCoordinateGenerator。

几何体类元素的子元素用来作为几何体类元素的子元素嵌套，辅助生成几何体。如**Error! Reference source not found.**节将使用到Coordinate为PointSet、IndexedLineSet、IndexedFaceSet等元素定义坐标点，有了坐标点后将坐标点依次连接起来就能形成任意复杂度的几何体。

* 1. 几何体（Geometry）类元素的外观（Appearance）类元素

Appearance、FillProperties、 ImageTexture、LineProperties、Material、MovieTexture、MultiTexture、MultiTextureTransform、PixelTexture、TextureTransform。

几何体类元素的外观类元素用来辅助几何体元素获得外观。默认生成的几何体表面都是默认的白色，通过外观类元素为几何体表面定义颜色、透明度，或者粘贴图像作为材质等，就可获得任意复杂美观的外观。四、节将使用Appearance、Material、ImageTexture、TextureTransform等元素为几何体定义外观。

* 1. 子（Child）类元素

Background 、Billboard、BooleanFilter、BooleanSequencer、BooleanToggle、BooleanTrigger、Collision、ESPDUTransform、Fog、GeoLocation、GeoLOD、GeoViewpoint、Group、HAnimHumanoid、IntegerSequencer、ntegerTrigger、LOD 、NavigationInfo 、 NurbsSet、Script 、 Shape 、StaticGroup 、 Switch 、TextureBackground 、WorldInfo。

子元素类的元素比较丰富，包括几何体与外貌结合、成组、场景背景、脚本程序等，本章主要在**Error! Reference source not found.**节使用到Shape元素将几何体与外貌结合。

* 1. 光照（Light）类元素

SpotLight 、PointLight 、DirectionalLight

X3D用来对真实世界进行虚拟现实模拟，真实世界的一切都依靠光照来照亮，不然就是漆黑一片，十、节将使用SpotLight定义光源。

* 1. 视点（Viewpoint）类元素

Viewpoint

X3D设计为把X3D浏览者当作X3D图形世界中的一个视点，该视点的位置、视角等可以变换而获得不同的X3D图形观察效果。十一、节将使用Viewpoint 变换观察视角。

* 1. 动画（Animation）类元素

TimeSensor、TimeTrigger、TouchSensor、PlaneSensor、SphereSensor 、ProximitySensor、StringSensor、CylinderSensor、GeoTouchSensor、KeySensor、LoadSensor、VisibilitySensor；NurbsSurfaceInterpolator 、OrientationInterpolator、PositionInterpolator 、PositionInterpolator2D 、ScalarInterpolator、ColorInterpolator、CoordinateInterpolator 、CoordinateInterpolator2D、GeoPositionInterpolator、NormalInterpolator 、NurbsOrientationInterpolator、 NurbsPositionInterpolator ；ROUTER；

动画类元素主要通过插值器、传感器、传送器获得动画效果，七、节将使用TimeSensor 、PositionInterpolator等接点获得三维动画。

* 1. 外部媒体引用（External Reference）类元素

Sound、AudioClip；Inline

X3D自然也必须支持声音、外部X3D等的引用，本章将在九、节使用Sound、AudioClip元素引用外部声音文件而获得立体声效。

* 1. 超链接（Hyperlink）类元素

Anchor

X3D自然也必须支持超链接，十二、节将使用Anchor元素定义超链接。

* 1. 其他元素

Contour2D、 ContourPolyline2D、GeoMetadata、GeoOrigin、HAnimDisplacer 、HAnimJoint 、HAnimSegment、HAnimSite 、MetadataDouble 、MetadataFloat 、MetadataInteger 、MetadataSet 、 MetadataString 、NurbsCurve2D 、 NurbsTextureCoordinate。

其他元素类的元素比较专门化，如面向地理模拟的元素等，将在专家系统领域应用广泛。

1. **X3D文档样式与CSS**

X3D当前一般不选用CSS样式技术（参见**Error! Reference source not found.**节、**Error! Reference source not found.**节），而通过外观(Appearance)类元素及其特性获得样式，所以X3D文档中不含有外部CSS引用、内部CSS、嵌入式CSS等内容。

1. **X3D文档逻辑结构与DOM**

上述BasicStructure.x3d文档DOM的逻辑关系将如图 6.3‑81所示。X3D DOM可被程序语言（如Jscript、C#等）动态操作，从而实现强大的X3D应用，将在具体涉及时详述。



图 6.3‑81. BasicStructure.x3d文档DOM图解

1. **HTML的格式标准：CSS**
2. **HTML的编程标准：DOM与JS\C#**
3. **X3D布局**

X3D文档属于视觉媒体，视觉媒体布局基本原理参见**Error! Reference source not found.**节，而且X3D属于三维图形动画视觉媒体，所以又具有面向三维图形动画的一些特征，如三维坐标系统等(参见**Error! Reference source not found.**)，二维媒体软件中常用的z轴方向的层也直接以z轴坐标实现了。X3D是三维的，所以浏览X3D时，浏览者(通过X3D浏览器，如BS Contact)可获得身临其镜的感觉，可在三维系统中任意方向导航漫游，所以X3D没有文档尺寸的概念，如果X3D签入在其他文档中（典型地如嵌入在HTML文档中，参见四、节），其尺寸则是嵌入时作为对象而设置的尺寸。浏览者默认处于（0，0，10）的位置朝Z轴负方向并穿过原点的方向观察场景并可变换视角(将在十一、节详述)。

X3D描述真实三维形体世界，是面向三维图形的定位机制，即三维的绝对定位为默认定位机制。一个X3D文档中，通过X3D元素生成三维对象，然后三维对象通过三维坐标系统布局，形成三维场景。X3D三维坐标系统布局主要是坐标变换，常用的坐标变换有平移（translate）、缩放（scale）、旋转（rotate）三种，通过元素<Transform>的特性指定。下面是一个变换的示例(参见示例文件Transform.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform><!--本行代码中的Transform是文档中的第一个Transform元素，是根坐标系--> 7. <Shape> 8. <Cylinder radius="0.25" height="7"/><!--生成园柱体--> 9. </Shape> 10. <Transform rotation="0 0 1 1.575"><!--本行代码中的Transform元素在其父元素（上一个Transform元素的坐标系基础上进行变换，平移了(3,0,0), 绕（0,0,1)轴即Z-轴旋转了1.575弧度（即约90度）--> 11. <Shape> 12. <Cylinder radius="0.25" height="7"/><!--生成园柱体--> 13. </Shape> 14. </Transform> 15. </Transform> 16. </Scene> 17. </X3D> |

Transform.x3d的运行结果如图 6.3‑82所示：

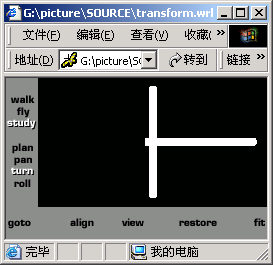


图 6.3‑82. Transform.x3d的运行结果

X3D文档由许多元素组成，X3D作为三维应用，其元素占位与二维的HTML元素、HTML元素占位机制有所不同，没有padding（衬距）、border（边框）、margin（边白）的概念(参见**Error! Reference source not found.**节)，只需考虑元素内容自身的占位。

思考： 从文档尺寸、坐标系统、布局系统、流式定位、相对定位、绝对定位、元素占位等方面，思考X3D布局机制与HTML布局机制(参见**Error! Reference source not found.**节、九、节)、XAML布局机制(参见**Error! Reference source not found.**节)的异同。

NURBS的全称是Non-Uniform Rational B-Spline（非均一有理B样条曲线），是三维形体建模的工业标准，这和路径是二维形状描述的工业标准是类似的，非常遗憾，多数X3D浏览器尚不支持X3D的NURBS元素，所以创作X3D时只好以点、线、面元素来近似获得NURBS元素的功能。在此主要理解三维形体构建的基本原理，复杂的三维形体必须借助软件辅助设计开发，将在**Error! Reference source not found.**详述Maya编辑三维形体。

任何几何体都可理解为是一组多边形面（最简单为三角形面）包围而成的，而多边形面又是由线段依次联接而定义的，线段又是由点联接定义的，也就是说任何几何体都是由点、线、面构成的，掌握点、线、面相关的元素将有利于几何体的生成。X3D 2.0 生成点、线、面的相关元素有PointSet,IndexedLineSet,IndexedFaceSet等。

下面是生成点集的一个示例(参见示例文件PointSet.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <PointSet> 9. <Coordinate point="0 0 0, 0 4 0,0 -1.33 -3.77,-3.26 -1.33 1.88,3.26 -1.33 1.88" /><!--根据四个坐标生成了四个点--> 10. </PointSet> 11. </Shape> 12. </Transform> 13. </Scene> 14. </X3D> |

注：这里的点是数学意义的点，包括后面将介绍的线和面也是数学意义上的线和面，即点没有大小、线没有粗细、面没有厚度。

下面是在点集的基础上按照点的顺序生成索引线集的一个示例(参见示例文件IndexedLineSet.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <IndexedLineSet coordIndex="0 1 -1,0 2 -1,0 3 -1,0 4 -1,1 2 -1,1 3 -1,1 4 -1,2 3 -1,3 4 -1, 4 2 -1" ><!--将生成的点的索引值，点根据索引值而互相连接起来--> 9. <Coordinate point="0 0 0, 0 4 0,0 -1.33 -3.77,-3.26 -1.33 1.88,3.26 -1.33 1.88" /><!--根据五个坐标生成了五个点--> 10. </IndexedLineSet> 11. </Shape> 12. </Transform> 13. </Scene> 14. </X3D> |

IndexedLineSet.x3d的运行结果如图 6.3‑83所示：

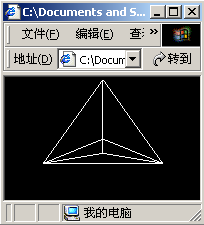


图 6.3‑83. IndexedLineSet.x3d的运行结果

下面是在点集的基础上按照点的顺序生成索引面集的一个示例(参见示例文件IndexedFaceSet.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <IndexedFaceSet coordIndex="1 4 2 -1,1 2 3 -1,1 3 4 -1,2 4 3 -1" ><!--将生成的点的索引值，点根据索引值而互相连接起来并形成面--> 9. <Coordinate point="0 0 0, 0 4 0,0 -1.33 -3.77,-3.26 -1.33 1.88,3.26 -1.33 1.88" /><!--根据五个坐标生成了五个点--> 10. <Color color="1 1 1,1 1 0,1 0 1,1 0 0,0 0 0" /><!--为面赋予颜色, X3D中的颜色使用归一化的RGB[[0.,1.], [0.,1.], [0., 1.]] ，其中1相当于非归一化的255。--> 11. </IndexedFaceSet> 12. </Shape> 13. </Transform> 14. </Scene> 15. </X3D> |

IndexedFaceSet.x3d的运行结果如图 6.3‑84所示：

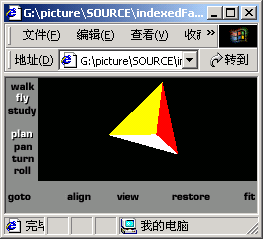


图 6.3‑84. IndexedFaceSet.x3d的运行结果

需注意的是每个面法线的方向与面的显示有关。面的法线方向是由点的顺序按照右手规则决定的。例如，决定一个面的点集序号为0、1、2、3，把右手握成拳，拇指伸开，其余四指指向0-1-2-3-0的顺序，则拇指的指向就是法线方向。如果想让索引面集的每一个面都可见，那么必须保证所有面的法线方向都是指向索引面集的外面。

注：几何意义上，IndexedFaceSet是基于多边形面来描述自由光滑曲面的，只要点取得足够多，就一定能够无限地逼近物体的真实形状，但这样在X3D文档中必须描述曲面上各个点的位置数据，将显著地增加文件的大小。X3D将引入NURBS元素，NURBS是三维造型中曲面设计与造型的工业标准，是一种用曲线和曲面表示轮廓和形体的方法，NURBS取代IndexedFaceSet造型将显著地优化X3D应用。

1. **X3D三维图形动画：NURBS形体、基本形体**

基本形体是指数学规律上较简单的几何体，如球体，方体、圆椎体等。X3D包含下列描述基本几何体的元素:<Sphere>、<Box>、<Cone>等。在此以<Box>为例进行叙述，<Sphere>、<Cone>可参见X3D标准中的说明自主上机实验。下面是以<Box>元素生成方体的一个示例(参见示例文件Box.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <Box /><!--生成立体体--> 9. </Shape> 10. </Transform> 11. </Scene> 12. </X3D> |

思考：使用IndexedFaceSet生成一个方体和使用Box生成一个方体的异同。

1. **X3D三维图形动画：形体的外貌**

X3D图形编辑的基本原理是先生成几何体，然后为几何体赋予材料或进行贴图来获得几何体的颜色、透明度、图案等外观属性。

1. **X3D三维图形动画：形体的外貌：材料**

X3D的材料是指几何体表面的材料感，包含了颜色、透明度、发光性等属性(参见**Error! Reference source not found.**节)，通过Material元素来定义。

下面的示例是为几何体外貌赋予材料后得到一个淡绿色的球(参见示例文件Material.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <Appearance> 9. <Material emissiveColor="0.4 1 0.3" /><!--生成漫射色作为球体的材料，x3d中颜色值以归一化的rgb方式表示--> 10. </Appearance> 11. <Sphere /><!--生成球体--> 12. </Shape> 13. </Transform> 14. </Scene> 15. </X3D> |

Material.x3d的运行结果如图 6.3‑85所示：



图 6.3‑85. Material.x3d的运行结果

1. **X3D三维图形动画：形体的外貌贴图**

通过为几何体赋予材料属性可以满足比较单一的外观属性要求，如均一的颜色，透明度等，当几何体的外貌要求复杂的的花纹、图案时，则需要通过贴图的机制，贴图是指在几何体的表面粘贴一层预定义的图案来获得外貌属性，这通过<Texture>元素引用外部图形、图像、视频等文件来实现静态或动态的贴图。下面是贴图的一个示例(参见示例文件Texture.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <Sphere /><!--生成球体--> 9. <Appearance> 10. <ImageTexture url="wwwEdu.jpg" /><!--引用同路径下的JPG图像文件贴图--> 11. </Appearance> 13. </Shape> 14. </Transform> 15. </Scene> 16. </X3D> |

Texture.x3d的运行结果如图 6.3‑86所示：

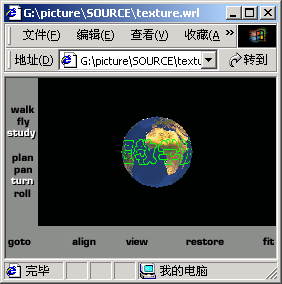


图 6.3‑86. Texture.x3d的运行结果

思考：X3D贴图与XAML图案、背景的比较。

默认的贴图方式（如示例中）下，图形图像是被拉伸或缩小到正好覆盖几何体表面上的，有时候为了得到其他材图粘贴效果，需要对贴图的坐标进行变换，贴图坐标的变换通过TextureTransform元素来定义，下面是一个示例(参见示例文件TextureTransform.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <Sphere /><!--生成球体--> 9. <Appearance> 10. <ImageTexture url="wwwEdu.jpg" /><!--引用同路径下的JPG图像文件贴图--> 11. <TextureTransform rotation="0.75" /><!--贴图坐标变换--> 12. </Appearance> 14. </Shape> 15. </Transform> 16. </Scene> 17. </X3D> |

TextureTransform.x3d的运行结果如图 6.3‑87所示：

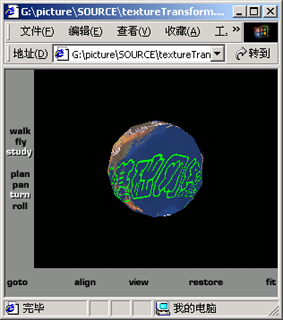


图 6.3‑87. TextureTransform.x3d的运行结果

TextureTransform元素定义一个贴图坐标系的二维变换，以获得所希望的图形图像在几何体表面上的粘贴方式。贴图坐标系是二维的，首先对贴图坐标系进行变换，变换后获得的当前贴图坐标系用于贴图映射。TextureTransform元素的变换与几何体局部坐标系的变换相互不影响。

1. **X3D三维图形动画：插值运算形成动画**

X3D动画支持持续帧动画和内插帧动画，在此主要叙述内插帧动画，理解内插帧动画的基本原理，复杂的动画必须借助软件辅助设计开发，将在**Error! Reference source not found.**详述Maya编辑三维动画。

X3D支持随时间改变元素的exposedAttribute类型的特性的特性值而获得内插帧动画效果。下面是X3D动画的基本原理和动画生成的基本方法。

1. **定义一个时间传感器元素。**

下面是TimeSensor元素的完整句法：

<TimeSensor

exposedAttribute SFString DEF " "

exposedAttribute SFTime cycleInterval 1 # (0,)



exposedAttribute SFBool enabled TRUE

exposedAttribute SFBool loop FALSE

exposedAttribute SFTime startTime 0 # (-,)



exposedAttribute SFTime stopTime 0 # (-,)



eventOut SFTime cycleTime

eventOut SFFloat fraction\_changed # [0,1]

eventOut SFBool isActive

eventOut SFTime time

>

TimeSensor定义一个时间传感器，该传感器随时间的流逝触发事件，所以具有驱动动画等功能。DEF特性指定传感器的名称，以便按名称调用。cycleInterval特性指定每个时间周期的长度。enabled特性指定该时间元素是否启用。loop特性指定当TimeSensor中指定的一个时间周期结束后是否重复运行。当loop设置为FALSE时，TimeSensor仅仅运行cycleInterval中指定的一个时间周期（或在一个周期还未完成之前就到达stopTime中指定的时间值）；当loop设置为TRUE时，TimeSensor不断地运行，直到到达stopTime中指定的时间或enable特性被设置为FALSE的时候。startTime特性指定TimeSensor开始运行的时间值。stopTime特性指定TimeSensor结束的时间值。如果其值小于或等于startTime的值，则被忽略。TimeSensor元素运行时，isActive 事件出发送TRUE，停止运行时发送FALSE。在TimeSensor元素开始运行时或每一个新的时间循环的开始，cycleTime 事件出发送时间。 (用于与其他基于时间的对象同步)time 事件出可在需要的时候发送时间。fraction\_changed事件出基于当前周期已经过的百分比发送值，该值是介于[0,1]之间的分数。例如在当前周期的一开始发送0，结束时发送1，30%时发送0.3等等。

1. **定义一个插值器元素**

X3D常用的插值元素有：ColorInterpolator,CoordinateInterpolator, OrientationInterpolator, PositionInterpolator, ScalarInterpolator等， 它们的功能不同，针对不同的需要进行插值，但句法完全相似，下面以ScalarInterpolator为例进行句法说明：

<ScalarInterpolator

exposedAttribute SFString DEF " "

eventIn SFFloat set\_fraction # (-,)



exposedAttribute MFFloat key [ ] # (-,)



exposedAttribute MFFloat keyValue [ ] # (-,)



eventOut SFFloat value\_changed

>

ScalarInterpolator定义一个标量（非矢量，即SFFloat类型的值，如透明度、光强度、半径等）插值变化，将在TimeSensor的驱动下使标量值动态插值变化而获得动画效果。DEF特性指定传感器的名称，以便按名称调用。eventIn事件进接收TimeSensor的fraction\_changed 事件出发送的值（介于[0,1]之间），然后根据key特性和keyValue特性指定的值进行插值运算，运算结果由value\_changed事件出发送出去。

key特性指定插值关键帧时间比例列表，是介于[0,1]之间的浮点数。

keyValue特性指定值的列表（对于ScalarInterpolator来说就是标量值）。这个值列表必须一一对应于key特性中指定的关键帧时间比例列表。

1. **定义一个欲产生动画效果的元素（或元素组）**
2. **将时间传感器元素中指定的时间特性传送给插值器元素中指定的插值特性获得插值结果，插值器元素的插值结果传送给欲动画的元素的相应特性，由于特性值的随时间变化而产生动画效果**

Route元素句法如下，一般都是成对出现，一般放置在X3D文档的最后。

<ROUTE nTimeSensor.fraction\_changed nScalarInterpolator.set\_fraction />

<ROUTE nScalarInterpolator .value\_changed nTransparency.set\_translation />

下面是一个坐标插值元素（CoordinateInterpolator）在时间传感器（Timesensor）的驱动下的变形动画，可以形象地将正四面体结构与平面正方形结构进行比较(参见示例文件coordinateInterpolator.x3d)。

|  |
| --- |
| 1. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 2. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 3. <X3D version="3.0"> 4. <Scene> 5. <Transform> 6. <Shape><!--定义一个平面正方形--> 7. <IndexedLineSet coordIndex="0 1 -1,0 2 -1,0 3 -1,0 4 -1,1 2 -1,1 3 -1,1 4 -1,2 3 -1,3 4 -1, 4 2 -1" > 8. <Coordinate DEF="CH4\_coord" point="0 0 0, 0 4 0,0 -1.33 -3.77,-3.26 -1.33 1.88,3.26 -1.33 1.88" /> 9. </IndexedLineSet> 10. </Shape> 11. </Transform> 12. <!--下面的代码定义时间传感器和坐标插值器，以获得平面正方形变化为正四面体的动画效果。--> 13. <CoordinateInterpolator DEF="coordInterp\_CH4\_morph" key="0, .2, .8, 1" keyValue="0 0 0,-4 4 0,4 4 0,-4 -4 0,4 -4 0,0 0 0,-4 4 0,4 4 0,-4 -4 0,4 -4 0,0 0 0, 0 4 0,0 -1.33 -3.77,-3.26 -1.33 1.88,3.26 -1.33 1.88,0 0 0, 0 4 0,0 -1.33 -3.77,-3.26 -1.33 1.88,3.26 -1.33 1.88" /><!--定义坐标插值器--> 14. <TimeSensor DEF="timeSensor\_CH4\_morph" cycleInterval="10" loop="true"/><!--定义时间传感器--> 15. <ROUTE fromField="fraction\_changed" fromNode="timeSensor\_CH4\_morph" toField="set\_fraction" toNode="coordInterp\_CH4\_morph"/> <!--定义一个传送，将"timeSensor\_CH4\_morph"名称的时间传感器的时间插值传送给"coordInterp\_CH4\_morph"名称的插值器--> 16. <ROUTE fromField="value\_changed" fromNode="coordInterp\_CH4\_morph" toField="set\_point" toNode="CH4\_coord"/><!--定义一个传送，将"coordInterp\_CH4\_morph"名称的插值器的插值结果传送给"CH4\_coord"名称的几何体--> 17. </Scene> 18. </X3D> |

coordinateInterpolator.x3d的运行结果如图 6.3‑88所示。

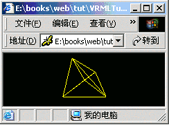
  

图 6.3‑88. coordinateInterpolator.x3d的运行结果

思考：X3D动画与XAML动画原理的异同。

1. **X3D文本**

可以在X3D场景中生成三维文本，通过<Text>元素生成。下面是一个三维文本的示例(参见示例文件Text.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"><!--声明文档架构--> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <Text string="Edu" /><!--X3D面向三维图形动画，而非面向文本，所以文本内容是作为<Text>元素的特性值而定义的，而面向文本的HTML中的文本是作为布局类元素的内容而定义的。BS Contact支持中文显示--> 9. </Shape> 10. </Transform> 11. </Scene> 12. </X3D> |

Text.x3d的运行结果如图 6.3‑89所示。X3D中的文本与其他三维图形元素的机制类似，所以适用于三维图形元素的特性，也同样适用于<text>元素，如三维坐标变换等。X3D中的文本可以看作是具有文本布局特性的形体，因此文本排版功能不及HTML等面向文本的机制，但更容易获得文本的三维形体化视觉效果。



图 6.3‑89. Text.x3d的运行结果

思考：从三维形体的角度思考X3D的NURBS、基本形体、文本之间的联系。

1. **X3D图像**

HTML设计为引用外部的图像文件到当前文档中，并布局。

X3D可以直接定义三维图形动画、文本，从媒体完整性的视角，还可能需要引用图像、视频、二维图形动画、三维图形动画（即外部X3D文件）、波形音频、MIDI器乐等外部媒体；从功能性的视角，还可能需要引用功能性组件。

X3D可以直接定义三维图形、文本，同时可以引用集成外部图像、图形、视频、动画、波形音频、MIDI器乐等已保存为文件的媒体，此外X3D文档支持引用已保存为文件的其他X3D文档，这样有利于X3D文档的扩展式重复使用。

前面叙述的贴图等功能事实上都是X3D外部媒体应用的实例，此外，X3D支持引用外部音频为X3D文档生成的三维图形世界增加立体声音世界。音频的引用是通过Sound和AudioClip实现的。下面是一个示例(参见示例文件Audio.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Shape> 8. <Sphere /><!--生成球体--> 9. </Shape> 10. <Sound> 11. <AudioClip url="sample.wav"/><!--引用同路径下的wav音频文件播放立体效果的声音--> 12. </Sound> 13. </Transform> 14. </Scene> 15. </X3D> |

1. **X3D光照**

X3D用来对真实世界进行虚拟现实模拟，真实世界的一切都依靠光照来照亮，不然就是漆黑一片，X3D文档在浏览时实际上拥有默认的光照，同时X3D还提供了DirectionalLight、PointLight、SpotLight等元素来模拟平行光、点光源、聚光灯等效果，例如SpotLight元素的完整句法如下：

<SpotLight

exposedAttribute SFFloat ambientIntensity 0 # [0,1] #光的漫射强度

exposedAttribute SFVec3f attenuation 1 0 0 # [0,] #衰减



exposedAttribute SFFloat beamWidth 1.570796 # (0,/2) #光柱宽度



exposedAttribute SFColor color 1 1 1 # [0,1] #光颜色

exposedAttribute SFFloat cutOffAngle 0.785398 # (0,/2) #光的切角



exposedAttribute SFVec3f direction 0 0 -1 # (-,) #光方向



exposedAttribute SFFloat intensity 1 # [0,1] #光强度

exposedAttribute SFVec3f location 0 0 0 # (-,) #光源位置



exposedAttribute SFBool on TRUE #光是否打开

exposedAttribute SFFloat radius 100 # [0,] #光半径



>

SpotLight元素定义了一个从指定点、沿着指定方向的锥形光，锥形光的方向、衰减等效果如图 6.3‑90所示：

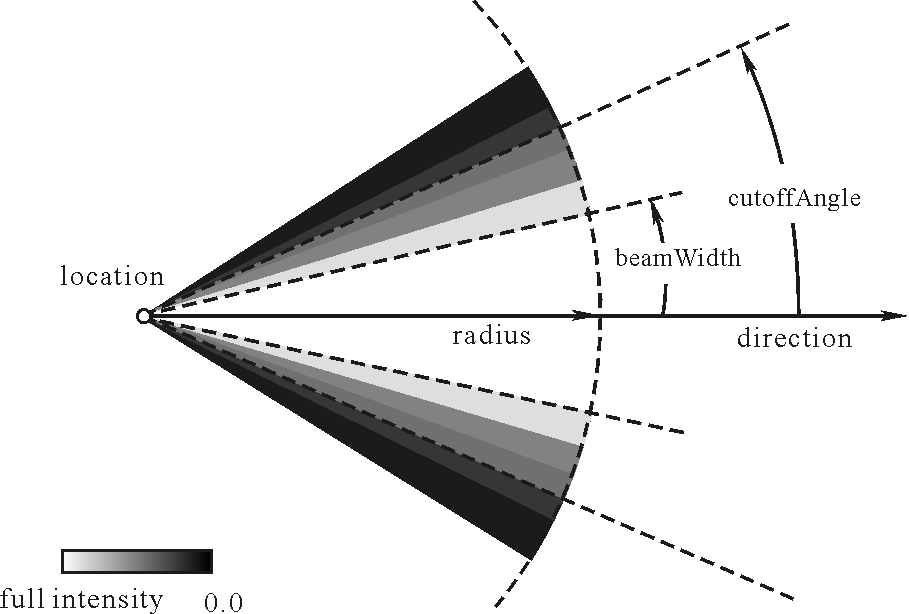


图 6.3‑90. SpotLight元素定义的聚光灯效果

1. **X3D视点**

X3D用来对真实世界进行虚拟现实模拟，X3D浏览者在X3D生成的三维图形世界中具有身临其境的感觉，这是因为X3D设计为把X3D浏览者当作X3D图形世界中的一个视点，该视点的位置、视角等可以变换而获得不同的X3D图形观察效果，视点通过Viewpoint元素定义，下面是Viewpoint的完整句法：

<Viewpoint

eventIn SFBool set\_bind

exposedAttribute SFFloat fieldOfView 0.785398 # (0,)



exposedAttribute SFBool jump TRUE

exposedAttribute SFRotation orientation 0 0 1 0 # [-1,1],(-,)



exposedAttribute SFVec3f position 0 0 10 # (-,)



field SFString description ""

eventOut SFBool isBound

>

Viewpoint 元素是可绑定元素。set\_bind事件进用来动态绑定该元素，即使该视点起作用。isBound事件出用来动态告诉X3D浏览器该视点是否正在起作用。

position和orientation 特性指定观察者的位置和方向。fieldOfView 特性指定一个视角。其值介于（0，）之间。jump 特性指定在不同的视点之间切换时浏览者的观察效果是突变的还是渐变的。“jump”是突变方式的，在起点至终点之间也不进行碰撞等检察。description 特性为该Viewpoint元素指定一个文本描述作为用户界面的提示作用。



Viewpoint 元素在局部坐标系中定义一个观察者的视点。视点预定义的默认位置在世界坐标系的（0 0 10），默认观察方向为（0 0 1 0）（向z-轴正方向即朝显示器屏幕向里观察），视点可以重新定义而获得不同的观察位置和方向，下面是一个示例(参见示例文件viewpoint.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Viewpoint position="0 0 10" orientation="0 0 1 0" description="front" /><!--命名为"front"的视点--> 7. <Viewpoint position="0 0 20" orientation="0 0 1 3.14" description="back" /><!--命名为"back"的视点--> 8. <Transform> 9. <Shape> 10. <Sphere /><!--生成球体--> 11. <Appearance> 12. <ImageTexture url="wwwEdu.jpg" /><!--引用同路径下的JPG图像文件贴图--> 13. </Appearance> 15. </Shape> 16. </Transform> 17. </Scene> 18. </X3D> |

viewpoint.x3d的运行结果如图 6.3‑91所示：

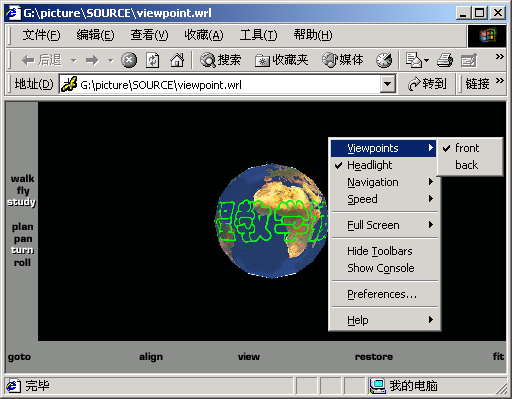


图 6.3‑91. viewpoint.x3d的运行结果

可以在“front”，“back”两个视点之间切换，体会不同的视点观察到的图形的不同效果。

1. **X3D超链接**

作为Web三维图形引用，X3D支持超链接的功能，下面是超链接的一个示例(参见示例文件Hyperlink.x3d)。

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> 2. <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN" 3. "http://www.web3d.org/specifications/x3d-3.0.dtd"> 4. <X3D version="3.0"> 5. <Scene> 6. <Transform> 7. <Anchor url="http://www.tup.com.cn"><!--定义超链接--> 8. <Shape> 9. <Box /> 10. </Shape> 11. </Anchor> 12. </Transform> 13. </Scene> 14. </X3D> |

一个X3D应用中往往含有多个X3D文件，其中一个是启动文件。通过一个文件启动之后，可以有两种机制将这些X3D文件连结起来。一是各个文件之间通过超链接互相跳转链接在一起，此时的信息交互机制是面向用户事件互动式驱动的，例如，用户通过鼠标单击超链接(即鼠标事件)、按键盘(即键盘事件)而交互等，这与使用三维图形界面软件类似；二是X3D的动画机制可随时间自动变换所交互的信息，例如用户观看一个线性X3D动画，这与观看视频类似。当然X3D信息交互也可以是上述两种的综合，所以X3D从媒体的角度可称为超三维图形动画媒体，X3D编辑者则可在超链接(事件)驱动为主导还是动画(时间)驱动为主导之间作出选择。

思考：从事件驱动与时间驱动的角度，思考X3D与XAML(参见**Error! Reference source not found.**节)、HTML(参见九、节)媒体交互性的异同。