http://dx.doi.org/10.5392/IJoC.2016.12.2.001

Specification and Implementation of Projective Texturing Node in X3D

In-Kwon Kim, Ho-Wook Jang, Kwan-Hee Yoo
Department of Digital Informatics and Convergence
Chungbuk National University, 1 Chungdaero, Seowongu, Cheongju, Chungbuk 361-763 Republic of Korea

Jong-Sung Ha
Department of Game and Contents
Woosuk University, 490 Hujongri, Samrae-Up, Wanju-Kun, Chonbuk 565-701 Republic of Korea

ABSTRACT

Extensible 3D (X3D) is the ISO standard for defining 3D interactive web- and broadcast-based 3D content integrated with multimedia. With the advent of this integration of interactive 3D graphics into the web, users can easily produce 3D scenes within web contents. Even though there are diverse texture nodes in X3D, projective textures are not provided. We enable X3D to provide SingularProjectiveTexture and MultiProjectiveTexture nodes by materializing independent nodes of projector nodes for a singular projector and multi-projector. Our approach takes the creation of an independent projective texture node instead of Kamburelis's method, which requires inconvenient and duplicated specifications of two nodes, ImageTexture and Texture Coordinate.

Key Words: *Projective Texture Mapping, Shadow Mapping, X3D, Scene Graph.*

I. INTRODUCTION

The Extensible 3D (X3D) [1]-[7] has been studied actively and it is being used in diverse 3D application services such as games, movies, education, advertisements in the Internet. The features of X3D 3D graphics include polygonal geometry, parametric geometry, hierarchical transformations, lighting, materials and multi-pass/multi-stage texture mapping; they are specified with Shape/Geometry3D component making figures, AppearanceNode describing the characteristic of the light, MaterialNode for the color of the material, and Texture component supporting textures. The texture component provides many TextureNodes: ImageTexture, MovieTexture, MultiTexture, MultiTextureCoordinate, MultiTextureTransform, PixelTexture, TextureCoordinate, TextureCoordinateGenerator, TextureProperties, and TextureTransform.

Even though there are many texture features, none of these supports the effects of projective texturing. Projective texture mapping (PTM) lets the texture image be projected in the projective volume seen from the particular point called 3D scene's projection point. It was studied by Everitt [8], and then Kamburelis [10] materialized the projective texture with the original progress of materializing shadow mapping. His method is inconvenient because the users had to add separate images through the ImageTexture node and give the effect of projective texture by using the Projected TextureCoordinate node.

* *Corresponding author, Email: khyoo@chungbuk.ac.kr*
Manuscript received Mar. 10, 2016; revised May. 23, 2016; accepted Jun. 02, 2016

In this paper, we propose a specification method for an independent projective texture node which combines both ImageAdding and Coordinate functions together. This new node is implemented in the environments of OpenGL and FreeWRL open source X3D viewer [15].

This paper consists of five sections. Section 2 discusses related work to the projective texture. Section 3 explains a newly proposed specification for projective texture nodes and the implementation results are described in Section 4. Finally, we summarize the overall results of our paper and discuss further research.

2. RELATED STUDIES

Projective texture mapping lets the texture image be projected in the projection volume with a view from a certain spot called 3D scene projection point. Projection volumes are chosen by projection parameters such as projection point, projection direction, and projection aspect ratio. This method is explained by Everitt [8] and Segal [12] in the process of texture mapping. Fig. 1 is a simple example of projective texture. Texture image can be projected and be seen on the 3D space when a projector is installed to the 3D space and a texture image is laid in front of it.

In order to understand the projective texture mapping, prior knowledge of the texture's coordinate formation is needed. Normally in the case of 2D textures, the texture mapping is done by establishing the texture coordinate value of 0 to 1 to the vertex of the material where the texture will be applied. However, instead of producing texture coordinate number as in the case of 2D Textures, projection texture mapping works in a way which a single projector is given to a space and the color that comes out from the projector reflects to the material.

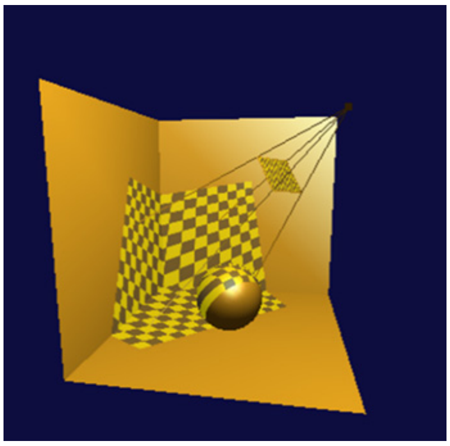
Fig. 1. Projective texturing

Kamburelis [10] studied and performed the functions on expanding nodes for shadows and projection textures in X3D. In his research, nodes were expanded and implemented to perform shadow effects and projection texture effects on the virtual space of X3D. He added certain fields to the original X3DLightNode and gave it the role of the projector, and expanded X3DTextureNode, X3DTextureCoorinateNode to create GeneratedShadowMapNode and ProjectedTextureCoordinateNode. And also he added fields like projectionNear, projectionFar into LightNode to carry out the functions of a projector. Fig. 2 is an example file that used the nodes which they have declared, and the example screen is shown on Fig. 3.

DEF redSpotlightSpotLight{
  direction 0 -1 0
  color 1 0.75 0.75
  ambientIntensity 0.1
  cutOffAngle 1.25
  beamWidth 1.25
  defaultShadowMapGeneratedShadowMap{
    update "ALWAYS"
    size 1024
  }
}
	DEF room Shape{
	  Appearance Appearance{
	    texture DEF spotLightTextureImageTexture{
	  url"image.jpg"
	  }
	}
	texCoord DEF projTexCoordProjectedCoordinate{
	projector USE redSpotLight
	}

(a)	(b)

Fig. 2. (a) Projector declared through SpotLightNode
(b) Definition of projective texture by using AppearanceNode

As we look at (a) in Fig. 2, projector is declared by the SpotLightNode, and (b) defined the AppearanceNode of the figure which receives the declared SpotLight. If we look at the Fig. 2, ImageTextureNode needs to be declared in order to receive ProjectionTextureImage, and ProjectedTexureCoordinateNode needs to be declared in order to get the projection texture coordinates. This inconveniency implies that this cannot be referred to as an independent ProjectionTextureNode. In this paper, we have expanded and implemented an independent ProjectionTexture and ProjectorNode in order to overcome the limitations of Kamburelis' research [10].

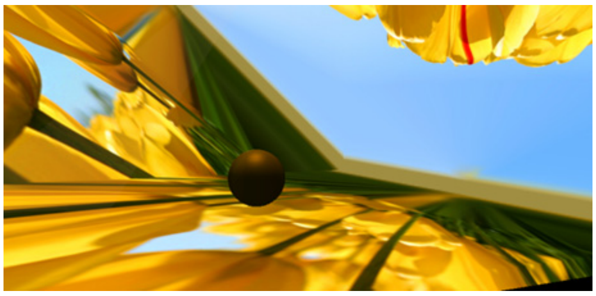
Fig. 3. Projection texturing under a spot light

3. X3D EXTENSIONFOR SPECIFYING PROJECTIVE TEXTURE

X3D specification defines 129 nodes, and these nodes have a hierarchy structure as seen as in Fig. 4. Every node in X3D derives from X3D Node, and each X3D Node consists of child nodes by its functions. X3D is formed as a hierarchy structure of child nodes and the actual performance is given out by the last node. The ProjectiveTextureNode and Projector Node, which we are going to expand, should not go against the hierarchy structures. Therefore, we expanded the Projective TextureNode as the child node of the Appearance Node, and expanded the ProjectorNode as the child node of ChildNode as shown in shading rectangle in Fig. 4.

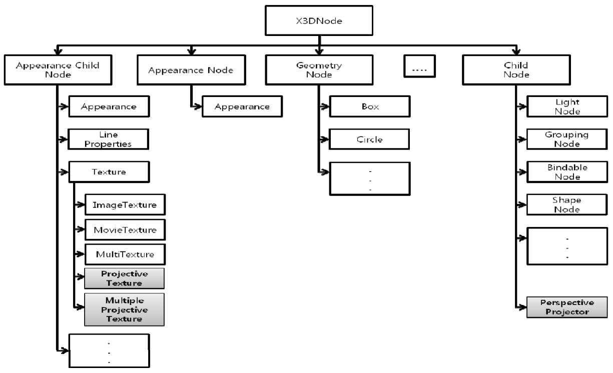

Fig. 4. Extended X3D hierarchy structure for projective texturing

In order to solve the problem appeared in results of Kamburelis, we need a node level of X3DTextureNode instead of X3DTextureCoordinateNode for independent projective texture node. Therefore, we have defines the following projective texture node as shown in Fig. 5.

First, the ProjectiveTexture node works as a projector on the virtual 3D space. The description field of this node tells the name of the projector, and makes the division of different projectors possible. Next, we added centerOfProjection, direction, aspectRatio, fieldofView, nearFar, upVector in order to create ProjectionVolume and ProjectionTextureCoordinate. First, centerOfProjection field shows the position of the projector, and this implies projection point. Direction is the way the projector is heading, and this implies to projection direction.

```
ProjectiveTexture : X3DchildNode{
  SFNode    [in,out]    metadata      NULL [X3DmetadataObject]
  SFString  [in,out]    description   “”
  SFVec3f   [in,out]    centerOfProjection    0 0 1
  SFVec3f   [in,out]    direction     0 0 1
  SFFloat   [in,out]    fieldOfView 45
  SFFloat   [in,out]    aspectRatio   1
  MFFloat   [in,out]    nearFar       1 10
  SFVec3f   [in,out]    upVector      0 1 0
}
```

Fig. 5. ProjectiveTexture node

The fieldOfView is the projector’s field of view, and aspectRatio is the aspect ratio of the width and length which refers to projection aspect ratio. Lastly, nearFar is the minimum and maximum distance that is shown on the screen. We could draw out ProjectionVolume and ProjectionTextureCoordinate from the added fields. This node is an independent ProjectiveTexture node which we can get ProjectiveTextureImage from, and also can express projective textures with the projector that we mentioned before. In this node, ProjectorName field lets a certain projector’s ProjectiveTexureCoordinate be brought through the name of the projector node. valueField helps the ProjectiveTexture to appropriately act functions of On/Off. Lastly, URL shows the images of the projector.

Fig. 6 is a MultiProjectiveTexture node which applies multiple projective textures on 3D objects, and this is a node expansion in order to apply more complex visual effects. Original nodes of X3D were made by citing MultiTextureNodes. The functions of each field are the same as the MultiTextureNode’s functions. The alpha field represents the degree of the original Alpha in the selection of mode field. colorField represents the original RGB colors in the selection of modes, and function field offers additional functions after the mode is applied. modeField establish multiple factor values for the mixture of color and image. The number of applied images matches with the number of modes. The source field is the field to decide the second factor’s color, and texture field includes ProjectiveTexture node. We can add projective nodes with this field.

```
MultipleProjectiveTexture: X3DtextureNode {
  SFFloat     [in,out] alpha          1 [0,1]
  SFColor     [in,out] color          1 1 1 [0,1]
  MFString    [in,out] function       []
  SFNode      [in,out] metadata       NULL [X3DmetadataObject]
  MFString    [in,out] mode           []
  MFString    [in,out] source         []
  MFNode      [in,out] texture [] [X3DtextureNode]
}
```

Fig. 6. MultipleProjectiveTexture node

4. IMPLEMENTATION RESULTS

We used the FreeWRL [15] in order to implement the functions of the proposed nodes. FreeWRL is an X3D open source viewer implemented by using OpenGL shader language (abbreviated as GLSL) which is based on C program. The FreeWRL uses PERL to help expand and generate our proposed X3D nodes easily. In order to do it, first, we read the X3D script file and do parsing hierarchical nodes X3D including our proposed nodes, ProjectiveTexture and MultiProjectiveTexture. And then, data are made up as a tree, based on the parsed data. The parsed tree structure data is saved in global structure. The saved data can get drawn as a 3D scene by the shader program, and here in FreeWRL, the shader programming is done by using GLSL. Data like vertex coordinates is processed in vertex shader, color information and texture is processed in FragmentShader, and lastly the 3D Scene is drawn based on the parsed data through GLSL.

By looking at Fig. 7, we can figure out that the name of PerspectiveTexture is ‘pt1’, and that the position of it is (X=3, Y=3, Z=3), the direction which pt1 views to is (X=1, Y=0, Z=1), field of view of the projector is 15, aspect ratio is 1, maximum distance of 10 and minimum distance of 1, and lastly, that the upVector is set to axis y. With this information, we can get the projection volume and projective texture coordinate. If we look at the appearance part of the Shape, we can see that the “pt1”projector is being designated through the projectorName field by ProjectiveTexture node, and sets the On/Off function value as true, and gets ProjectiveTextureimages. Here we give “apple.jpg” as a projective texture image. By these explained nodes, we could expand our own ProjectiveTexture node. Fig. 7 (b) shows the projective texture mapping result by using FreeWRL for specified X3D file shown in Fig. 7 (a).

```
<X3D profile="Interactive" version="3.3">
<Scene>
<ProjectiveTexture
      description='pt1' centerOfProjection='3 3 3'
      direction='-1 0 -1' fieldOfView='15'
aspectRatio='1' nearFar='1 10' upVector='0 1 0'   />
<Shape>
<Appearance>
<Material diffuseColor='1 1 1'/>
<ProjectiveTexture value='true' projectorName='pt1'   url=’apple.jpg'/>
</Appearance>
<IndexedFaceSet solid='false' coordIndex= "3 2 1 0 -1, 4 5 2 3-1, 5 6 1 2 -1">
<Coordinate point="1 0 1, -1 0 1, -1 0 -1, 1 0 -1, 1 1 -1, -1 1 -1, -1 1 1 "/>
</IndexedFaceSet>
</Shape>
</Scene>
</X3D>
```

(a) A X3D file including proposed projective texture node

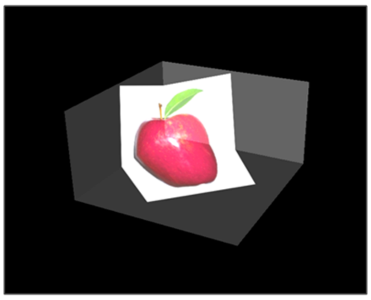
(b) A screen shot of FreeWRL browsing result for the X3D file of (a)

Fig. 7. An example of the newly expanded ProjectiveTexture node into X3D

Fig. 8 shows a multiple projective texture mapping result of a X3D file specified according to proposed node definition. Fig. 8 (a) show a X3D file including a specification of multiple projective textures with three each projective textures defined as illustrated in Fig. 7. As shown in Fig. 8 (a), each of three projective textures has all the same specification except for only both of different directions and texture images. Different directions of the three projective textures are “(0,0,-1)”, “(-2,0,-1)”, and “(-1, 0, -1)”, respectively, and texture image files are “ChessSet.jpg”, “apple.jpg”, and “Chess.jpg”. Fig. 8 (b) shows the projective texture mapping result by using FreeWRL for specified X3D file shown in Fig. 8(a).

```
<X3D profile="Interactive" version="3.3">
<Scene>
<ProjectiveTexture
      description='pt1' centerOfProjection='3 3 3'
      direction='0    0 -1' fieldOfView='15'
aspectRatio='1' nearFar='1 10' upVector='0 1 0'    />
<ProjectiveTexture
      description='pt2' centerOfProjection='3 3 3'
      direction='-2    0 -1' fieldOfView='15'
aspectRatio='1' nearFar='1 10' upVector='0 1 0'    />
<ProjectiveTexture
      description='pt3' centerOfProjection='3 3 3'
      direction='-1 0 -1' fieldOfView='15'
aspectRatio='1' nearFar='1 10' upVector='0 1 0'    />
<Shape>
<Appearance>
<Material diffuseColor='1 1 1'/>
<MultipleProjectiveTexture>
               <ProjectiveTexture value='true'projectorName='pt1'
                 url=’ChessSet.jpg'/>
               <ProjectiveTexture value='true'projectorName='pt2'
                 url=’apple.jpg'/>
               <ProjectiveTexture value='true'projectorName='pt3'
                 url=’Chess.jpg'/>
              </MultipleProjectiveTexture>
</Appearance>
<IndexedFaceSet solid='false' coordIndex=
                 "3 2 1 0 -1, 4 5 2 3-1, 5 6 1 2 -1">
<Coordinate point="1 0 1, -1 0 1, -1 0 -1, 1 0 -1, 1 1 -1,
                  -1 1 -1, -1 1 1 "/>
</IndexedFaceSet>
</Shape>
</Scene>
</X3D>
```

(a) A X3D file including proposed multiple projective texture node

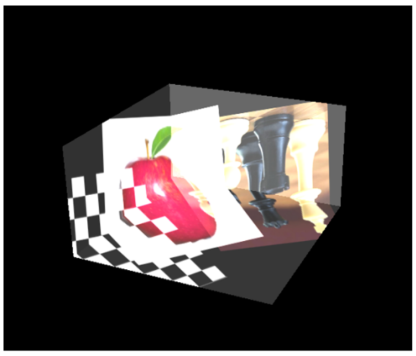
(b) A screen shot of FreeWRL browsing result for the X3D file of (a)

Fig. 8: An example of multiple projective texture mapping results by using MultipleProjectiveTexture nodes

5. CONCLUSION

In this paper, we specified and implemented a new feature of X3D, which is called Projective Texture. Our implementation is based on the parameters of other X3D nodes with GLSL shaders. This research resolved the inconvenient necessity of the previous method [10] for separately creating ImageTexture and ProjectedTexture Coordinate nodes to get image effects for projective texturing. In the future, we will enhance the inconvenience of adding the projective texture to each appearance node of multiple objects. This implementation also will be able to stretch out to open source viewers like X3Dom [16] from the current situation of focusing to X3D viewers like FreeWRL.

ACKNOWLEDGEMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2014R1A1A2055379) and by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2016-H8501-16-1013) supervised by the IITP(Institute for Information & communication Technology Promotion).

REFERENCES

[1] Web3D CONSORTIUM, “VRML” http://web3d.org/x3d/specifications, Mar., 2016,
[2] WEB 3D CONSORTIUM, “X3D” http://web3d.org/x3d/specifications, Mar., 2016,
[3] WEB 3D CONSORTIUM, “X3D texture” http://web3d.org/ x3d/ specifications, Mar., 2016,
[4] WEB 3D CONSORTIUM, “X3D 3D Texturing Component,” http://web3d.org/x3d/specifications, Mar., 2016,
[5] X3D Specification, X3DPublicSpecifications\ISO-IEC-19775-1.2-X3D-AbstractSpecification, http://www.web3d.org, Mar., 2016.

[6] X3D: Extensible 3D Graphics for Web Authors, http://x3d graphics.com/, Mar., 2016.
[7] DonBrutzman and Leonard Daly, X3D: Extensible 3D Graphics for Web Authors, Morgan Kaufmann, 2007.
[8] Cass Everitt, Projective Texture mapping, nVidia, 2001.
[9] J. R. Spann and K. S. Kaufman, "Photogrammetry using 3D Graphics and Projective Textures," IAPRS, vol. XXXIII, Part B5/1, Com. V, Amsterdam, pp. 748-755, 2000.
[10] Michalis Kamburelis, "Shadow maps and projective texturing in X3D," the 15th International Conference on Web 3D Technology, pp. 17-26, 2010.
[11] Eunjung Kim, Kwan-Hee Yoo, Je-Hoon Lee, Yong-Dae Kim, and Younggap You, "Composite Endoscope Images from Massive Inner Interestine Photos," Lecture Notes on Artificial Intelligence 4570, pp. 1042-1051, 2007.
[12] Mark Segal, Carl Korobkin, Rolf van Widenfelt, Jim Foran, and Paul Haeberli "Fast shadows and lighting effects using texture mapping," 19th Annual Conference on Computer graphics and interactive techniques, pp. 249-252, Jul., 1992.
[13] In-Kwon Kim, Ho-Wook Jang, and Kwan-HeeYoo, "Definition of Projective Texture Mapping Node in X3D," In Proc. of the Korea Computer Graphics Society, vol. 21, no.3, pp. 83-84, Jul., 2015.
[14] http://www.users.globalnet.co.uk/~tomh, Mar., 2016.
[15] FreeWRL, "X3D Viewer", http://freewrl.sourceforge.net/, Mar., 2016.
[16] X3Dom, http://www.x3dom.org, Mar., 2016.

In-Kwon Kim
He is a student who is working for department Digital Informatics and Convergence at Chungbuk National University, Korea. He received the B.E. in computer of computer education at Chungbuk National University, Korea in 2014, and also received M.S. in digital informatics and convergence from Chungbuk National University, Korea in 2016. His research interests include computer graphics, CAD/CAM, and 3D digital contents.

Ho-Wook Jang
He is a PhD Candidate who is working for Department of Digital Information and Convergence at Chungbuk National University, Korea, and also a principal member of research staff who is working for Department of Next Generation Content Research Division at ETRI (Electronics and Telecommunications Research Institute), Korea. He received the B.E. in computer engineering from Kyungpook National University, Korea in 1986, and also received M.S. in computer science from KAIST (Korea Advanced Institute of Science and Technology), Korea in 1988. His research interests include computer graphics, 3D character animation, 3D digital contents

Jon-Sung Ha
He is a professor who is working for department Game and Contents at Woosuk University, Korea. He received the B.E. in computer of computer engineering at Seoul National University, Korea in 1984, and also received M.S., Ph.D. in computer science from KAIST (Korea Advanced Institute of Science and Technology), Korea in 1986 and 1996, respectively. His research interests include computational geometry, computer graphics, CAD/CAM, and 3D digital contents.

Kwan-Hee Yoo
He is a professor who is working for department of Computer Science at Chungbuk National University, Korea. He received the B.S. in computer Science from Chonbuk National University, Korea in 1985, and also received M.S., Ph.D. in computer science from KAIST (Korea Advanced Institute of Science and Technology), Korea in 1988 and 1995, respectively. His research interests include u-Learning system, computer graphics, 3D character animation, dental/medical applications.