17.2.2 Lighting model

17.2.2.1 Introduction

The X3D lighting model provides detailed equations that specify the colours to apply to each geometric object. For each object, the values of the Material, color, and/or texture currently being applied to the object are combined with the lights illuminating the object and the currently bound *X3DFogObject* (if specified). These equations are designed to simulate the physical properties of light striking a surface.

If a programmable shader is defined for an Appearance node, the lighting model shall be disabled and replaced by the functionality implemented by the shader program. See 31 Programmable shaders component for more information.

17.2.2.2 Lighting 'off'

A Shape node is unlit if either of the following is true:

a. The shape's *appearance* field is `NULL` (default).
b. The *material* field in the Appearance node is `NULL` (default).

NOTE Geometry nodes that represent lines or points do not support lighting.

If the shape is unlit, the colour (I_{rgb}) and alpha (A, 1−transparency) of the shape at each point on the shape's geometry is specified in Table 17.2.

Table 17.2 — Unlit colour and alpha mapping

Texture type	Colour per-vertex or per-face	Colour NULL
No texture	$I_{rgb} = I_{Crgb}$ $A = 1$	$I_{rgb} = (1, 1, 1)$ $A = 1$
Intensity (one-component)	$I_{rgb} = I_T \times I_{Crgb}$ $A = 1$	$I_{rgb} = (I_T, I_T, I_T)$ $A = 1$
Intensity+Alpha (two-component)	$I_{rgb} = I_T \times I_{Crgb}$ $A = A_T$	$I_{rgb} = (I_T, I_T, I_T)$ $A = A_T$
RGB (three-component)	$I_{rgb} = I_{Trgb}$ $A = 1$	$I_{rgb} = I_{Trgb}$ $A = 1$

RGBA (four-component)	$I_{rgb} = I_{Trgb}$ $A = A_T$	$I_{rgb} = I_{Trgb}$ $A = A_T$

where:

A_T = normalized [0, 1] alpha value from 2 or 4 component texture image

I_{Crgb} = interpolated per-vertex colour, or per-face colour, from Color node

I_T = normalized [0, 1] intensity from 1 or 2 component texture image

I_{Trgb} = colour from 3-4 component texture image

17.2.2.3 Lighting 'on'

If the Shape node is lit (*i.e.*, a Material and an Appearance node are specified for the Shape), the Material and Texture nodes determine the diffuse colour for the lighting equation as specified in Table 17.3.

The Material's diffuseColor field modulates the color in the texture. Hence, a diffuseColor of white will result in the pure color of the texture, while a diffuseColor of black will result in a black diffuse factor regardless of the texture.

The Material's transparency field modulates the alpha in the texture. Hence, a transparency of 0 will result in an alpha equal to that of the texture. A transparency of 1 will result in an alpha of 0 regardless of the value in the texture.

Table 17.3 — Lit colour and alpha mapping

Texture type	Colour per-vertex or per-face	Color node NULL
No texture	$O_{Drgb} = I_{Crgb}$ $A = 1\text{-}T_M$	$O_{Drgb} = I_{Drgb}$ $A = 1\text{-}T_M$
Intensity texture (one-component)	$O_{Drgb} = I_T \times I_{Crgb}$ $A = 1\text{-}T_M$	$O_{Drgb} = I_T \times I_{Drgb}$ $A = 1\text{-}T_M$
Intensity+Alpha texture (two-component)	$O_{Drgb} = I_T \times I_{Crgb}$ $A = A_T$	$O_{Drgb} = I_T \times I_{Drgb}$ $A = A_T$
RGB texture (three-component)	$O_{Drgb} = I_{Trgb}$ $A = 1\text{-}T_M$	$O_{Drgb} = I_{Trgb}$ $A = 1\text{-}T_M$

RGBA texture (four-component)	$O_{Drgb} = I_{Trgb}$ $A = A_T$	$O_{Drgb} = I_{Trgb}$ $A = A_T$

where:

I_{Drgb} = material *diffuseColor*
O_{Drgb} = diffuse factor, used in lighting equations below
T_M = material *transparency*

All other terms are as defined in 17.2.2.2 Lighting off.

17.2.2.4 Lighting equations

An ideal X3D implementation will evaluate the following lighting equation at each point on a lit surface. RGB intensities at each point on a geometry (I_{rgb}) are given by:

$$I_{rgb} = I_{Frgb} \times (1 - f_0) + f_0 \times (O_{E\,rgb} + \mathrm{SUM}(\, on_i \times attenuation_i \times spot_i \times I_{Lrgb} \times (ambient_i + diffuse_i + specular_i)))$$

where:

$attenuation_i = 1 / \max(c_1 + c_2 \times d_L + c_3 \times d_L^2 , 1)$
$ambient_i = I_{ia} \times O_{Drgb} \times O_a$

$diffuse_i = I_i \times O_{Drgb} \times (\mathbf{N} \cdot \mathbf{L})$
$specular_i = I_i \times O_{Srgb} \times (\mathbf{N} \cdot ((\mathbf{L} + \mathbf{v}) / |\mathbf{L} + \mathbf{v}|))^{shininess \times 128}$

and:

$\cdot$ = modified vector dot product:
if dot product < 0, then 0.0, otherwise, dot product
c_1 , c_2, c_3 = light i *attenuation*
d_V = distance from point on geometry to viewer's position, in coordinate system of current fog node
d_L = distance from light to point on geometry, in light's coordinate system
f_0 = fog interpolant, see Table 17.5 for calculation
I_{Frgb} = currently bound fog's *color*

I_{Lrgb} = light i *color*

I_i = light i *intensity*

I_{ia} = light i *ambientIntensity*

L = (PointLight/SpotLight) normalized vector from point on geometry to light source i position

L = (DirectionalLight) -direction of light source i

N = normalized normal vector at this point on geometry (interpolated from vertex normals specified in a node derived from *X3DNormalNode* or calculated by browser)

O_a = *X3DMaterialNode ambientIntensity*

O_{Drgb} = diffuse colour, from a node derived from *X3DMaterialNode*, a node derived from *X3DColorNode*, and/or a texture node

O_{Ergb} = *X3DMaterialNode emissiveColor*

O_{Srgb} = *X3DMaterialNode specularColor*

on_i = 1, if light source i affects this point on the geometry,

0, if light source i does not affect this geometry. The following conditions indicate that light source i does not affect this geometry:

a. if the geometry is farther away than *radius* for PointLight or SpotLight;

b. if the geometry is outside the enclosing *X3DGroupingNode*; and/or

c. if the *on* field is FALSE.

shininess = *X3DMaterialNode shininess*

spotAngle = arccosine($-\mathbf{L} \cdot \mathbf{spotDir}_i$)

$spot_{BW}$ = SpotLight i beamWidth

$spot_{CO}$ = SpotLight i *cutOffAngle*

$spot_i$ = spotlight factor, see Table 17.4 for calculation

$\mathbf{spotDir}_i$ = normalized SpotLight i *direction*

SUM: sum over all light sources i

v = normalized vector from point on geometry to viewer's position

Table 17.4 — Calculation of the spotlight factor

Condition (in order)	$spot_i$ =
$light_i$ is PointLight or DirectionalLight	1
spotAngle ≥ $spot_{CO}$	0

spotAngle ≤ $spot_{BW}$	1
$spot_{BW}$ < spotAngle < $spot_{CO}$	(spotAngle - $spot_{CO}$) / ($spot_{BW}$ - $spot_{CO}$)

Table 17.5 — Calculation of the fog interpolant

Condition	f_0 =
no fog	1
fogType "LINEAR", d_V < fogVisibility	(fogVisibility-d_V) / fogVisibility
fogType "LINEAR", $d_V \geq$ fogVisibility	0
fogType "EXPONENTIAL", d_V < fogVisibility	exp(-d_V / (fogVisibility-d_V))
fogType "EXPONENTIAL", $d_V \geq$ fogVisibility	0