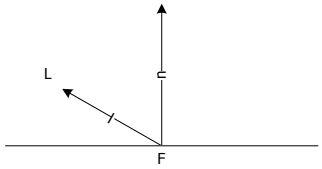
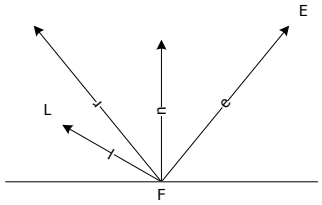
Diffuse Lambert



L = Light Position
F = Fragment Position
n = Normal Vector

l = L – F
x = dot(normalize(l), normalize(n))

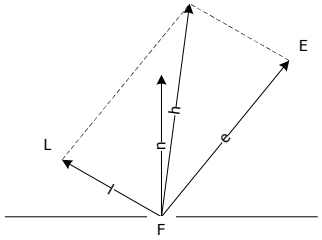
Specular Phong



L = Light Position
E = Eye Position
F = Fragment Position
n = Normal Vector

l = L – F
e = E - F
r = -reflect(e, n)
x = dot(normalize(r), normalize(l))

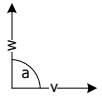
Specular Blinn



L = Light Position
E = Eye Position
F = Fragment Position
n = Normal Vector

l = L – F
e = E - F
h = l + e
x = dot(normalize(h), normalize(n))

dot(v, w)



dot(v, w) = cos(a) * length(v) * length(w)

length(v) = 1
length(w) = 1

dot(v, w) = cos(90°) * 1 * 1 = 0.0

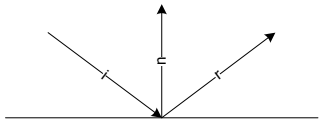
dot(v, w) = cos(0°) * 1 * 1 = 1.0

dot(v, w) = cos(180°) * 1 * 1 = -1.0

Spaces

Object Space (3D)
↓
Model Transformation
↓
World Space (3D)
↓
View Transformation
↓
Eye Space (3D)
↓
Projection Transformation
↓
Clip Space (3D)
↓
Project (Perspective Divide)
↓
Image Space? (2D)
↓
Viewerport Transformation
↓
Window/Screen/Viewport? Space (2D)

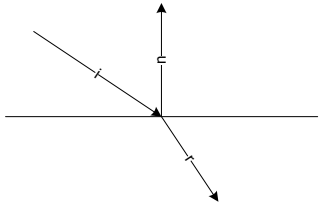
reflect(i, n)



i = Incident Vector
r = Reflection Vector
n = Normal Vector

r = reflect(i, n)
r = i - 2 * n * dot(n, i)

refract(i, n, eta)

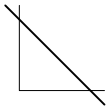


i = Incident Vector
r = Refraction Vector
n = Normal Vector

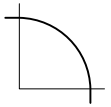
r = ?

Attenuation

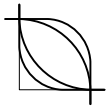
Linear Falloff = 1.0 – (d / maxd)



Quadratic Falloff = 1.0 - (d / maxd) ^ 2



Monomial Falloff = 1.0 – (d / maxd) ^n



Cartesian Coordinate System (Thumb=X, Index=Y, Middle=Z)

Left Hand / Z-Up

Right Hand / Z-Up

Panda3D

Right Hand / Y-Up

OpenGL