SIMLAB RT1.0

OFFICIAL USERS MANUAL

CONTENTS

MATERIALS **5**

Introduction 5
Dielectric Material 6
Usage 6
Sample Result 6
Parameters 7
Examples 7
Thin Dielectric Material 9
Usage 9
Sample Result 9
Parameters 9
Examples 10
Mirror Material 12
Usage 12
Sample Result 12
Parameters 13
Examples 13
Matte Material 14
Usage 14
Sample Result 14
Parameters 15
Examples 15
Metal Material 18
Usage 18
Sample Result 18
Parameters 18
Examples 19
Metallic Paint Material 23
Usage 23
Sample Result 23
Parameters 23
Examples 25
Shiny Metal Material 28
Usage 28
Sample Result 28
Parameters 29
Examples 29
Plastic Material 31
Usage 31
Sample Result 31
Parameters 32
Examples 34
Thin SSS Material 35

Usage 35
Sample Result 35
Parameters 35
Examples 36
VELVET MATERIAL 38
Usage 39
Sample Result 39
Parameters 39
Examples 40
EMITTER MATERIAL 43
Usage 44
Sample Result 44
Parameters 44

TEXTURES 45

INTRODUCTION 45
SUPPORTED FILE TYPES 45
TEXTURE COORDINATES 46
COMMON TEXTURE PROPERTIES 46
Texture map scale 46
Texture map offset 47
GENERIC TEXTURE MAP TYPES 48
Opacity map 48
Bump map 50
Normal map52

LIGHTS 55

INTRODUCTION 55
POINT LIGHT 55
Description 55
Sample Result 56
Parameters 56
DIRECTIONAL LIGHT 57
Description 57
Sample Result 57
Parameters 58
DISTANT LIGHT 59
Description 59
Sample Result 60
Parameters 60
Examples 61
SPOT LIGHT 61
Description 61
Sample Result 62
Parameters 62
AMBIENT LIGHT 63
Description 64
Sample Result 64

Parameters 64
ENVIRONMENT LIGHT 65
Description 65
Sample Result 65
Parameters 66
SKY LIGHT 67
Description 67
Sample Result 68
Parameters 68
Examples 69
EMITTER LIGHT 72
Description 72
Sample Result 72
Parameters 73

CAMERAS 73

INTRODUCTION 73
SUPPORTED CAMERA TYPES 74
COMMON CAMERA PROPERTIES 74
CAMERA DEPTH OF FIELD 75
PERSPECTIVE CAMERA PROPERTIES 77
MULTICAMERA FEATURE 77

BACKGROUND 78

INTRODUCTION 79
SPHERICAL BACKGROUND 79
Usage 80
Parameters 80
Examples 80
PLANAR BACKPLATE 82
Usage 83
Parameters 83
Examples 83
ENVIRONMENT MAP 84
Usage 84

GROUND OPTIONS 84

INTRODUCTION 85
GROUND SHADOWS 85
Usage 85
Parameters 85
Examples 86
GROUND REFLECTIONS 87
Usage 87
Parameters 87
Examples 88

TONEMAPPING 90

INTRODUCTION 90

SUPPORTED TONE MAPPING TYPES 90
LINEAR **ERROR! BOOKMARK NOT DEFINED.**
Parameters ***Error! Bookmark not defined.***
Examples ***Error! Bookmark not defined.***
REINHARD 91
Parameters *91*
UNCHARTED 2 91
Parameters *91*

MATERIALS

INTRODUCTION

SimLab RT supports physically based materials, with advanced parameters to allow both beginning and seasoned artists to get their desired photo-realistic results.

This section lists currently supported SimLab RT materials, gives an example of each material, its parameters, and usage of each parameter.

For each material parameter, the range that SimLab RT expects is listed in the parameters table. This allows anyone who wants to utilize SimLab RT as their rendering engine to understand the expected input, and what does it mean.

Appearance modeling is an active research area; SimLab RT follows latest trends to get best possible materials efficiency and accuracy, in addition to adding more materials with new releases.

The following materials are discussed in the following pages:

Dielectric

Thin Dielectric

Mirror

Matte

Metal

Metallic Paint

Shiny Metal

Plastic

Thin SSS

Velvet

Emitter

It is worth noting that default values can be set explicitly or implicitly just by ignoring the parameter altogether in the material definition. This means that any parameters that are not provided explicitly shall take their default values.

We always like to hear feedback about which parameters you feel are missing, and which materials should be added next. In addition we'd like to know which parameters deserve further illustration in this manual.

DIELECTRIC MATERIAL

USAGE

Dielectric material is used to simulate glass, transparent plastic, transparent liquids, and materials of similar nature.

A dielectric material has an index of refraction (called eta or IOR) which corresponds to the index of refraction of the material inside the object.

Since the surface of the object on which dielectric material is applied divides between two different indexes of refraction, it is called an "interface". So with a dielectric material, user has the ability to specify outside and inside index of refractions, where outside IOR maps to the one outside the object surface, and inside IOR is the one on the opposite side.

As an example, for surfaces interfacing air, etaOutside is 1, and etaInside is whatever IOR the material has, for glass that would be 1.5-1.65 (depending on glass type) and for water is 1.33. More specifically:

> If surface is a boundary between glass and air, need to set: etaOutside = 1.0 and etaInside = 1.5
> If surface is a boundary between glass and water, need to set: etaOutside = 1.5, etaInside = 1.33

Most objects will be interfacing air, so etaOutside is usually set to 1.

SAMPLE RESULT

The image to the right is generated with dielectric material and default parameters. Notice that if a parameter is not set explicitly, it will be automatically assigned the default value.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	***Type***	***Range***	***Default***	***Description***
MaterialEtaOutside	float	1.0f - 10.0f	1.0f	Index of refraction outside the dielectric interface
MaterialEtaInside	float	1.0f - 10.0f	1.4f	Index of refraction inside the dielectric interface
MaterialTransmission	(float,float,float)	{ 0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f,1.0f,1.0f	Transmission color inside the transparent object
MaterialRoughness	float	0.0f – 1.0f	0.0f	Controls the roughness of the dielectric material. Increasing roughness value results in more glossy (ie less specular) reflections and refractions. This effect is demonstrated in the examples section.

EXAMPLES

Following set of examples are rendered with varying transmission colors of the dielectric material

Example 1 - *Transmission: (0.8, 0.3, 0.8)*

Example 2 - *Transmission: (0.3, 0.8, 0.3)*

The following set of examples display the effect of roughness on the dielectric material

Example 3 – Roughness: 0.0
Transmission: (0.4, 0.4, 0.5)

Example 4 - Roughness: 0.1
Transmission: (0.4, 0.4, 0.5)

Example 5 – Roughness: 0.4
Transmission: (0.4, 0.4, 0.5)

THIN DIELECTRIC MATERIAL

USAGE

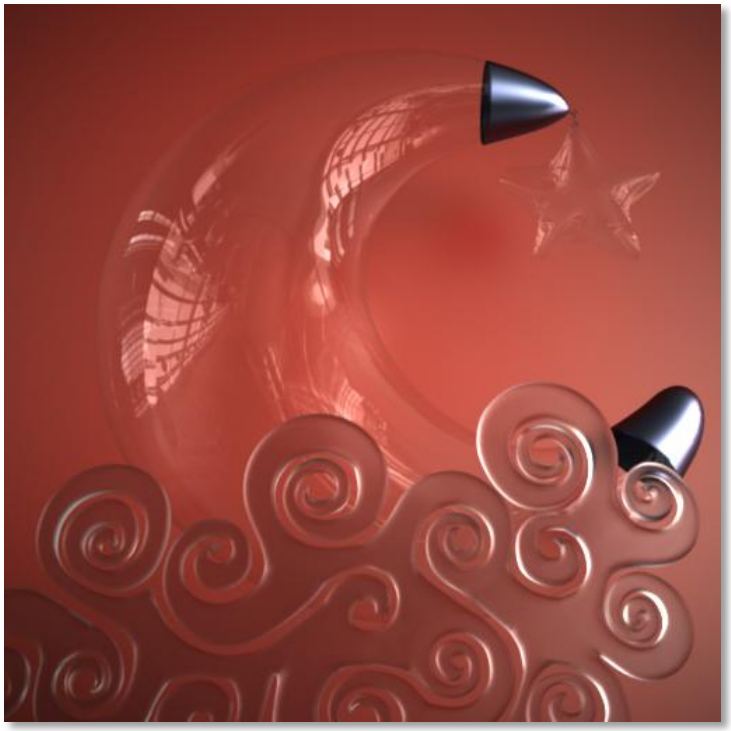
Thin dielectric material is used to approximate thin glass, by assuming that rays do not get refracted inside the glass. This material should, in general, be applied to an object with a single open boundary (i.e. a surface of glass which has no thickness).

Thin dielectric material is similar to a dielectric material in that it takes a transmission color and an index of refraction for the material. The provided IOR (or eta) is used only to calculate the Fresnel effect for reflection, and not to perform any actual refraction of rays, and outside IOR of the dielectric material is assumed to be 1.0f (ie air or vacuum). In addition, thin dielectric material takes a thickness parameter to simulate the decreased radiance of the ray as it passes through the medium.

SAMPLE RESULT

The image to the right is generated with thin dielectric material and default parameters. Notice that if a parameter is not set explicitly, it will be automatically assigned the default value.

Notice how the material appears to be made of a thin shell which is the distinctive property of this material compared to regular glass.



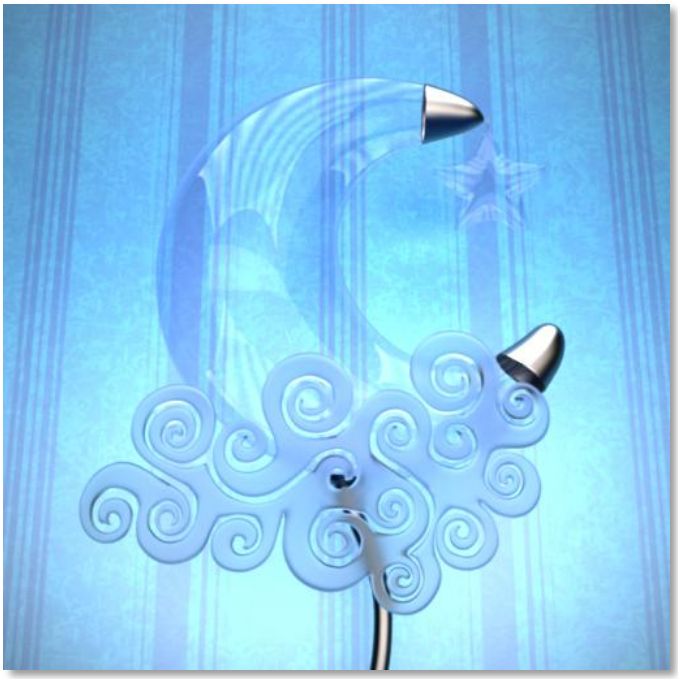
PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialEta	float	1.0f - 10.0f	1.4f	Index of refraction inside the dielectric interface
MaterialThickness	float	0.0f - inf	0.1f	Simulated thickness of the thin dielectric object (higher thickness means more attenuation of rays leaving the object)
MaterialTransmis-sion	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f,1.0f,1.0f	Transmission color inside the transparent object

EXAMPLES

Following examples are rendering with varying thin dielectric material thickness values. Notice in the following examples the lack of refraction inside the object when thin dielectric material is utilized



Example 1 – Thickness: 0.01

Example 2 – Thickness: 0.1

Example 3 – Thickness: 0.2

Example 4 – Thickness: 0.3

MIRROR MATERIAL

USAGE

As the name implies, mirror material simulates mirrors with optional reflectance color. It reflects incoming rays about the normal (which can be modulated through a bump or normal textures)

There is no need for an index of refraction for mirror material since all rays are reflected without regards to Fresnel's equation.

Reflectance of the mirror material is a color that determines the appearance of the material and modulates the reflected light.

SAMPLE RESULT

The image to the right is generated with mirror material and default parameters. Notice that if reflectance is not set explicitly, it will be automatically assigned the default value (that is, white color)

In the image below, the crescent mostly reflects the environment, and on the edges is reflecting the background object.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialReflectance	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f}	1.0f,1.0f,1.0f	Reflectance color to modulate reflected light

EXAMPLES

Following examples are rendered with varying reflectance colors.

Example 1 – Reflectance: (0.9, 0.1, 0.5)

Example 2 – Reflectance: (0.1, 0.5, 0.9)

MATTE MATERIAL

USAGE

Matte material (also known as Lambertian) is a diffuse material which reflects incoming light into all directions. It can be used to simulate various materials including walls, stones, rubber, non-reflective wood, and similar materials.

The most pronounced property of Matte material is the lack of specular or glossy reflections, so any real world material which lacks these types of reflections can be best simulated through matte material.

The sigma option of matte material controls how dusty the material looks. Low sigma values give rubbery feel, whereas high sigma gives dusty feeling to the material similar to stones and alike.

SAMPLE RESULT

The image to the right is generated with matte material and default parameters, with an addition of yellow reflectance texture that gives the materials its color.

Notice that if any parameter is not set explicitly, it will be automatically assigned the default value of that parameter.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialReflectance	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f, 1.0f, 1.0f	Reflectance color to modulate reflected light
MaterialMapReflectance	Texture		No Texture	A texture map to modulate reflected light instead of constant color
MaterialOffsetReflectance	(float,float)	{-inf to inf, -inf to inf }	0.0f, 0.0f	Offset of reflectance texture map
MaterialScaleReflectance	(float,float)	{-inf to inf, -inf to inf }	1.0f, 1.0f	Scale of reflectance texture map
MaterialSigma	float	0.0f - 1.0f	0.0f	Value to control the dusty look of the material (see examples)

EXAMPLES

Following examples are rendered with varying matte material sigma value.

Example 1 – Sigma: 0.0
Reflectance: (0.9, 0.6, 0.98)

Example 2 - Sigma: 0.5
Reflectance: (0.9, 0.6, 0.98)

Example 3 – Sigma: 1.0
Reflectance: (0.9, 0.6, 0.98)

METAL MATERIAL

USAGE

Metal material is used to generate metallic materials of various types, for the model to be truly physically based, it requires few parameters to be set by the user according to the desired metal.

Metallic materials have controllable roughness parameter, which controls the glossiness of the reflections. As an example, old metals have a higher roughness value as a result of aging, and therefore don't reflect as much as a new object with the same material.

The parameters which need to be physically set for metals include real and imaginary parts of the index of refraction, the examples section includes samples of these parameters and a method to obtain them for other metals. As can be seen in the examples section, utilizing these parameters can largely increase the believability of these metals, and is therefore recommended to use physical measurements. Another common method in case these measurements could not be obtained, is to start from a similar looking metal, and adjust reflectance color only to get closer to the desired result.

SAMPLE RESULT

The image to the right is generated with metal material and chrome metal parameters as described in the examples section (notice that default parameters correspond to Aluminum material):

Notice that if any parameter is not set explicitly, it will be automatically assigned the default value of that parameter.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialReflectance	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.90091f, 0.91537f, 0.92006f	Reflectance color to modulate reflected light
MaterialMapReflectance	Texture		No Texture	A texture map to modulate reflected light instead of constant color
MaterialOffsetReflectance	(float,float)	{-inf to inf, -inf to inf }	0.0f, 0.0f	Offset of reflectance texture map
MaterialScaleReflectance	(float,float)	{-inf to inf, -inf to inf }	1.0f, 1.0f	Scale of reflectance texture map
MaterialEta	(float,float,float)	{1.0f – 10.0f, 1.0f – 10.0f, 1.0f – 10.0f }	1.768f, 1.015192f, 0.72122f	Real part of index of refraction of metal, wave-length dependent
MaterialK	(float,float,float)	{0.0f – 10.0f, 0.0f – 10.0f, 0.0f – 10.0f }	7.9819f, 6.6273f, 5.7556f	Imaginary part of index of refraction of metal, also known as extinction coefficient, wavelength de-pendent
MaterialRoughness	float	0.0f - 1.0f	0.1f	Controls roughness of reflections, zero gives spec-ular reflections and one results in diffuse reflec-tions.

EXAMPLES

Notice that eta and k above are both wavelength dependent. It would be best to match these properties with physically measured values of the metal. To do that, notice that main wavelengths are:

Wavelength of red light : 680 nm (0.680 um)

Wavelength of green light : 550 nm (0.550 um)

Wavelength of blue light : 475 nm (0.475 um)

The wavelengths above can be entered into a chart or a tool that gives eta and k values of the desired material corresponding to the specified wavelength.

For example, the tool at refractiveindex.info gives the following values for Gold:

eta = 0.13544, 0.42415, 1.26175 for R,G, and B wavelengths

k = 3.8820, 2.4721, 1.8014 for R,G, and B wavelengths

reflectance = 0.96688, 0.79156, 0.39633 for R,G, and B wavelengths

The table below was generated in a similar fashion

Material	eta	k	reflectance
Gold	0.13544 0.42415 1.26175	3.8820 2.4721 1.8014	0.96688 0.79156 0.39633
Silver	0.14 0.125 0.13329	4.3695 3.339 2.7028	0.97254 0.95972 0.93793
Copper	0.2139 0.943918 1.15106	3.9646 2.5945 2.4926	0.95023 0.64076 0.57526
Aluminum	1.768 1.015192 0.72122	7.9819 6.6273 5.7556	0.90091 0.91537 0.92006
Platinum	2.47848 2.1313 1.90571	4.3888 3.7147 3.2893	0.68388 0.63882 0.60427
Chrome	3.06769 3.1724 2.54232	3.3605 3.3266 3.2521	0.55923 0.55436 0.56023

Following examples are rendered with varying metal material parameters taken from the list above, roughness values was chosen based on desired visual appearance

Example 1 – Gold Metal

- Eta: (0.13544, 0.42415, 1.26175)
- K: (3.8820, 2.4721, 1.8014)
- Reflectance: (0.96688, 0.79156, 0.39633)
- Roughness: 0.1

Example 2 – Silver Metal

- Eta: (0.14, 0.125, 0.13329)
- K: (4.3695, 3.339, 2.7028)
- Reflectance: (0.97254, 0.95972, 0.93793)
- Roughness: 0.1

Example 3 – Copper Metal

- Eta: (0.2139, 0.943918, 1.15106)
- K: (3.9646, 2.5945, 2.4926)
- Reflectance: (0.95023, 0.64076, 0.57526)
- Roughness: 0.3

METALLIC PAINT MATERIAL

USAGE

Metallic paint material is great for simulating materials with a colored layer coated by a dielectric layer. In addition, it has settings to add and control reflective flakes distribution within the dielectric layer.

According to the above, metallic paint material has lots of controls to make it such a powerful material. The settable parameters cover all three layers (base,coat, and flakes), where the flakes layer is optional and can be easily removed by setting flakes coverage to zero.

The coat layer has settings for color, depth (to control layer absorption), and index of refraction. Flakes layer has settings for coverage, color, and size. In addition the user has the ability to add variation to flakes depth and angle, so that not all flakes have the same depth or same angle from normal. Base layer has one settable parameter corresponding to its color.

It is worth mentioning that the flakes layer is generated by a smart proprietary algorithm that does not need to generate textures nor require complex calculations. As such, flakes are rendered with an efficient algorithm, requiring no addition storage or any texture coordinates to be applied to the object.

SAMPLE RESULT

The image to the right is generated with metallic paint material with default settings, notice that default color of metallic paint is set to red. Only flakes size parameter was modified to match object scale.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialCoatColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f}	0.95f, 0.95f, 0.95f	Color of the transparent coat, which affects the color of the light reaching the base layer
MaterialCoatThickness	float	0.f - inf	0.1f	Thickness of the coat layer, affecting light intensity reaching the base (see examples section)
MaterialCoatEta	float	1.0f - 10.0f	1.45f	Index of refraction of the coat layer
MaterialFlakesSize	float	0.f - inf	0.01f	Scale of the flakes in the flakes layer. Determined based on the desired world size of the flakes
MaterialFlakesColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f}	1.0f, 1.0f, 1.0f	Color the modulates light reflected off the flakes
MaterialFlakesCoverage	float	0.0f - 1.0f	0.1f	Controls the percentage of surface that is covered by flakes. For example, 0.2 means that 20% of surface is covered by flakes
MaterialFlakesDepthVariation	float	0.0f - 1.0f	1.0f	Controls the variation in depth of flakes, for example, setting this parameter to zero brings all flakes to the surface of the clear coat, whereas a depth variation of 1 means the flakes cover the whole range from clear coat surface down to base layer surface
MaterialFlakesAngleVariation	float	0.0f - 1.0f	1.0f	Controls the angles variation of flakes. Setting this to zero means all flakes will have same normals as surface, increasing this number means that flakes will have random angles with surface nor-

				mal
MaterialBaseColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f}	1.0f, 0.0f, 0.0f	Color of the base layer

EXAMPLES

The examples in this section are rendered with varying metallic paint material parameters. In the following set of examples notice how changing the size of the flakes affect the look of the glitter in the material

Example 1 – FlakesSize: 0.0001
baseColor: (0.1411, 0.2235, 0.8784)

Example 2 – FlakesSize: 0.001
baseColor: (0.1411, 0.2235, 0.8784)

In the following examples, notice how increasing the thickness of the coat layer darkens the light reflected off the material (both base and flakes look darker)

Example 1 – coatThikness: 0.1

Example 2 – coatThikness: 1.0

The other common properties in the previous renders are:

- coatColor: (0.95f, 0.95f, 0.95f)
- flakesSize: 0.0003
- baseColor: (0.8411, 0.2235, 0.8784)

The following example shows the effect of changing the flakes color, in addition, the flakes coverage is increased, so flakes cover 20% of object surface:

Example 1 – flakesColor: (1.0f, 0.9f, 0.0f)

Example 2 – flakesColor: (1.0f, 1.0f, 1.0f)

The other common properties in the previous renders are:

- coatColor: (0.95f, 0.95f, 0.95f)
- flakesSize: 0.0003
- baseColor: (0.8411, 0.2235, 0.8784)

SHINY METAL MATERIAL

USAGE

Shiny metal material can be used to simulate shiny materials in general, for example, this material is ideal for generating pearl material.

The shiny metal is a three layer material consisting of a diffuse base with settable color, mixed with a glitter glossy layer, and both are covered by a perfectly reflective layer.

The user can control the color of the base, color and glossiness of the glitter layer, and the index of refraction of the reflective layer, and therefore has a full control over the look of the material

SAMPLE RESULT

The image to the right is generated with shiny metal material with default settings, except with the shadeColor set to green.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialShadeColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.95f, 0.95f, 0.95f	Color of the base layer
MaterialEta	float	1.0f - 10.0f	1.45f	Index of refraction of the coat layer
MaterialGlitterColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f, 1.0f, 1.0f	Color of glossy layer
MaterialGlitter-Spread	float	0.0f - 1.0f	0.1f	Glossiness of the glossy layer (higher values mean increased roughness, lower values mean more specular reflection)

EXAMPLES

Following examples are rendered with varying shiny metal material parameters.

In the first set of examples notice the effect of changing glitter color on the overall look of the material. Notice how changing the giltter color only affects the material color to a certain degree, to control the overall color of the material, user needs to modify the shade color (i.e. the base layer color).

Example 1 – *G*litter disabled glitterColor: (0.0, 0.0, 0.0)

Example 2 - glitterColor: (0.65, 0.15, 0.0)

Example 3 – glitterColor: (0.0, 0.15, 0.65)

In the following examples, notice how increasing the glitter spread increases the glossiness of the glossy glitter layer

Example 1 – glitterSpread: 1.0

Example 2 – glitterSpread: 0.5

Example 3 – glitterSpread: 0.1

The other common properties in the previous renders are:

- shadeColor: (0.5f, 0.4f, 0.5f)
- glitterColor: (0.0, 0.15, 0.65)

PLASTIC MATERIAL

USAGE

Plastic material is used to simulate reflective non-metallic materials (including plastic, leather, and glossy materials in general). Notice that even though the name implies a specific kind of material, plastic material is actually rather general and can be used to simulate a wide range of non-metallic materials (indeed, even metallic materials can be nicely simulated when Fresnel reflections option is turned off. Still, utilizing the metallic material with all its physically based controls is the most accurate method for simulating metals).

The parameters of plastic material are straightforward and consist of pigment color, index of refraction, and roughness of the reflection. Increasing the roughness of plastic material makes the reflection more glossy, and the highest value of one makes the reflections appear totally diffused.

SAMPLE RESULT

The image to the right is generated with plastic material with default settings, notice that default color of metallic paint is set to red. Only flakes size parameter was modified to match object scale.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialPigmentColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.85f, 0.85f, 085f	Color of the plastic material
MaterialMapPigmentColor	Texture			Texture map to modulate plastic material color
MaterialOffsetPigmentColor	(float,float)	{-inf to inf, -inf to inf }	0.0f, 0.0f	UV offset of the pigment color texture map
MaterialScalePigmentColor	(float,float)	{-inf to inf, -inf to inf }	1.0f, 1.0f	UV scale of the pigment color texture map
MaterialReflectionColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.85f, 0.85f, 085f	Color of plastic material reflections, this modulates the reflected light bouncing off the object
MaterialEta	float	1.0f - 10.0f	1.45f	Index of refraction of the plastic material. Only utilized if Fresnel reflections are enabled.
MaterialEnableFresnel	bool	true or false	TRUE	Determines whether Fresnel reflections are enabled or disabled. For plastic materials, Fresnel reflections should be enabled, but it can be disabled to simulate metallic looking materials
MaterialRoughness	float	0.0f - 1.0f	0.01f	Roughness of the plastic material (see examples), higher value increases glossiness of the plastic material

EXAMPLES

In the following example notice how increasing roughness of plastic material increases glossiness of reflections, a value of 'one' would result in a diffuse reflection.

Example 1 – roughness: 0.0

Example 2 – roughness: 0.2

The example to the right shows the effect of disabling Fresnel reflections (default is enabled), which results in more metallic reflections were the amount of reflection does not fade away as the surface becomes perpendicular to the eye.

This can be visually noticed when compared to the render above, specially at the center of the crescent.

Example 3 – enableFresnel: false
reflectionColor: (0.05, 0.05, 0.1), roughness: 0.0

THIN SSS MATERIAL

USAGE

Thin SSS material is a specialized material for simulating thin materials which reflects, absorbs, and transmits different portions of light. This makes this material ideal for simulating lamp shades, tree leaves, and similar materials of thin objects.

It is worth noting that thin SSS material is applied to the surface only, so that no light interaction takes place inside the object itself.

Thin SSS material has a number of parameters to facilitate its functionality. In general, the user has the ability to control reflection, absorption, and transmission of light.

SAMPLE RESULT

The image to the right is generated with Thin SSS material with the following settings (again, we are talking about the material of the moon and star):

- *reflection_coeffecient = 1*
- *absorption_coeffecient = 0*
- *transmission_coeffecient = 0.65*
- *reflectance = white (1,1,1)*
- *translucence = yellow (1, 1, 0.1)*

Notice the effect of yellowish translucency on the objects. Notice also that transmission coefficient + reflection coefficient add up to more than one. If used without normalization, this would result in unrealistic material which breaks the low of conservation of energy. This is taken care of automatically by the renderer, and normalization takes place internally without informing the user.

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description

MaterialReflectionCoeffecient	float	0.0f - 1.0f	0.6f	The portion of light that is reflected off the object after modulation with reflectance color
MaterialAbsorptionCoeffecient	float	0.0f - 1.0f	0.05f	The portion of light that is absorbed by the object (ie. not reflected or transmitted)
MaterialTransmissionCoeffecient	float	0.0f - 1.0f	0.45f	The portion of light that is transmitted through the surface
MaterialReflectance	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f, 1.0f, 1.0f	Reflectance color that modulates reflected light bouncing off the object
MaterialMapReflectance	Texture		No Texture	A texture map to modulate reflected light instead of constant color
MaterialOffsetReflectance	(float,float)	{-inf to inf, -inf to inf }	0.0f, 0.0f	Offset of reflectance texture map
MaterialScaleReflectance	(float,float)	{-inf to inf, -inf to inf }	1.0f, 1.0f	Scale of reflectance texture map
MaterialTranslucence	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.5f, 0.5f, 1.0f	Translucence color that modulates light transmitted through the surface
MaterialSigma	float	0.0f - 1.0f	0.0f	Controls how dusty the object looks (please refer to sigma definition of Matte material, creates a similar effect here)

EXAMPLES

In the following examples notice the effect of modifying translucence color. Apart from the clear differences on the objects themselves, notice the subtle difference on shadow color of the moon and start objects between the two examples.

Example 1 – translucence: (1.0f, 0.5f, 1.0f)
(purple translucence)

Example 2 – translucence: (1.0f, 1.0f, 0.0f)
(yellow translucence)

The other common properties in the previous renders are:

- reflection_coeffecient: 0.5
- reflectance: (1.0f, 1.0f, 1.0f)
- reflectance map aaplied

In the following examples, notice the effect of increasing translucency level beyond reflection level

Example 3 – reflection_coeffecient 0.5
absorption_coeffecient 0.0
transmission_coeffecient 0.5

Example 4 – reflection_coeffecient 0.5
absorption_coeffecient 0.0
transmission_coeffecient 1.0

The other common properties in the previous renders are:

- reflectance: (1.0f, 1.0f, 1.0f)
- reflectance map aaplied
- translucence: (1.0f, 0.5f, 1.0f)

VELVET MATERIAL

USAGE

As its name implies, velvet material can be used to simulate velvet fabric, however it has the ability and controls to do more than that by providing a generic back-scattering BRDF framework that can be used to simulate other materials like general fabric, peach, moon, and dusty materials.

As such, velvet material has controls for the amount of backscattering (which gives a dusty look as shown in the examples section), in addition to a falloff property, which means the object can exhibit different colors if viewed straight-on compared to acute angles, again this property should be clarified in the examples section.

SAMPLE RESULT

The image to the right is generated with a velvet material with the following settings:

- *reflectance = red(0.5, 0.0, 0.0)*
- *backscattering = 0.5*
- *horizonScatteringFallOff = 2.0*
- *horizonScatteringColor = red(0.7, 0.2, 0.2)*

In this case we are giving slightly darker red (0.5,0.0,0.0) to straight on faces, and slightly brighter red (0.7,0.2,0.2) to the ones viewed with greater angles.
Backscattering is enabled with a relatively small value to simulate backscattering effect taking place in the velvet material (same effect would be seen on dusty objects and earth's moon)

PARAMETERS

(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
MaterialReflectance	(float,float,float)	{0.0f – 1.0f,	1.0f, 1.0f,	Reflectance color modulating light reflected off surface parts that are viewed

		0.0f – 1.0f, 0.0f – 1.0f }	1.0f	straight on
MaterialBackScattering	float	0.0f - 1.0f	0.6f	determines how much of the incident light is reflected in the direction where it came from, remaining portion of the light is scattered normally in the forward direction. This property explains why a full moon (where both eye and light come from same direction) appears significantly brighter than half moon, where light comes from a different direction than the eye
MaterialHorizonScatteringColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.5f, 0.5f, 1.0f	The color which the object takes as angle is increased between the viewer and the object
MaterialHorizonScatteringFallOff	float	0.0f - 1.0f	0.0f	Determines the Falloff speed to go from reflectance color for straight on faces, to horizontalScatteringColor for faces perpendicular to the viewer.

EXAMPLES

In the following examples notice how the horizon scattering color affects the material color specially where surface normal gets perpendicular to the eye.

Example 1 – horizonScatteringColor: (0.55f, 0.45f, 0.4f) *(grey falloff)*

Example 2 – horizonScatteringColor: (0.55f, 0.45f, 0.2f) *(yellow falloff)*

The other common properties in the previous renders are:

- reflectance: (0.32, 0.18, 0.14)
- backscattering: 0.5
- horizonScatteringFallOff: 3

next example shows the effect of Falloff factor of the velvet material. Notice that as value increases, the horizontalScatteringColor covers less area of the object. In other words, as the value is decreased, the horizontalScatteringColor starts taking effect at smaller angles, which in turn means larger coverage.

Example 3 – horizonScatteringFallOff: 5

Example 4 – horizonScatteringFallOff: 2

The other common properties in the previous renders are:

- reflectance: (0.32, 0.18, 0.14)
- backscattering: 0.5
- horizonScatteringColor: (0.55, 0.45, 0.4)

Last set of examples display the effect of varying the backScattering value, notice that as the value is increased, more light is reflected back in the light direction, and therefore less light is reflected to the eye (the camera) which results in darker color in surface that is facing the camera. Of course, if user was looking from direction of the light, we would have noticed higher brightness as backscattering is increased

Example 5 – backScattering: 0

Example 6 – backScattering: 0.5

Example 7 – backScattering: 1

Example 8 – backScattering: 2

All other parameters are the same in the renders above

EMITTER MATERIAL

USAGE

Emitter material is used to turn any 3d mesh into a light source, where the user can set the color and strength of the light.

SAMPLE RESULT

The image to the right displays how the crescent 3D object was turned into a mesh emitter for the whole scene just by applying a mesh emitter material to it.

Notice that an object with a mesh emitter material is also treated as a light in SimLab RT, therefore you can see further discussion on mesh emitter material when considered as a light source later in the lights section of this document.

PARAMETERS

Name	Type	Range	Default	Description
MaterialC	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f, 1.0f, 1.0f	Color of the mesh emitter material, controls the light color emitted from the object

MaterialStrength	float	0.0f - 1.0f	0.0f	The strength of the mesh emitter light, the higher the strength the stronger the light source

TEXTURES

INTRODUCTION

SimLab RT supports textures to modulate many effects, such as object's color. Different types of textures can be applied to the object at the same time, even though only one texture coordinates assignment is supported in the current release.

The materials section listed the specific supported textures for each of the materials. On top of that, there are texture maps that are generic to all materials and will be clarified in this section.

The goal of this section is to clarify the following parts of SimLab RT:

- Supported file types
- Texture coordinates
- Common texture properties
    - Scale of texture map
    - Offset of texture map
- Generic texture map types that can be applied to all materials
    - Opacity map
    - Bump map
    - Normal map

SUPPORTED FILE TYPES

All of the following image file types are supported by SimLab RT and can be used as texture maps:

- BMP
- EXR
- GIF
- HDR
- PPM
- PFM
- PNG
- PSD
- RAW

- TIFF

The above covers most common file formats used in computer graphics today, so the user should not have a problem saving and bringing any image file to SimLab RT.

TEXTURE COORDINATES

In order to apply a texture map to an object, the user needs to provide texture coordinates of the object on which the texture will be applied, the UV texture coordinates are well known in 3D computer graphics and need to be applied to each vertex of the mapped object. Applying a texture map to an object with no texture coordinates result in unexpected behavior.

COMMON TEXTURE PROPERTIES

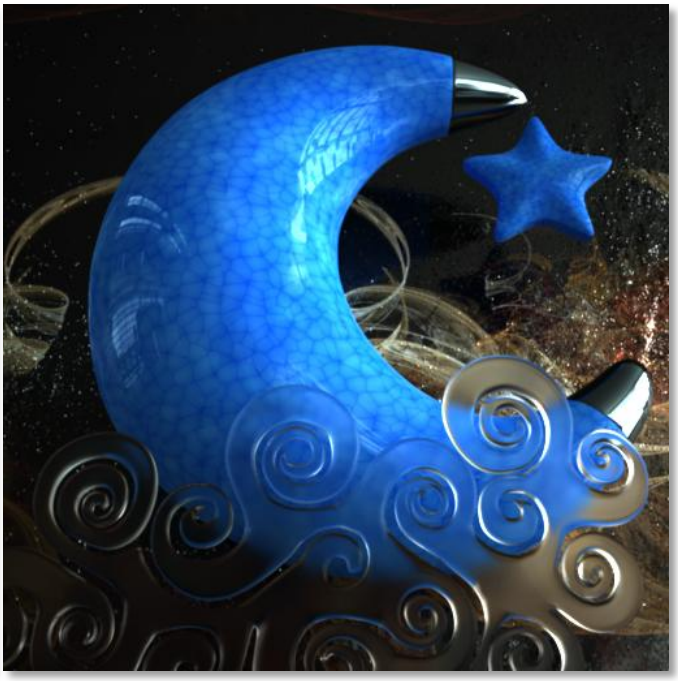
All texture maps share two major properties, which give the user more flexibility in controlling the look of the texture without the need of regenerating texture coordinates or modifying the mapped image itself.

TEXTURE MAP SCALE

The scale of texture map controls the repetition of the mapped image when the object UV coordinates change from 0 to 1. notice that the two scales (in the U and V directions) can be different and not necessarily the same. Changing the scale of one dimension relative to the other would affect the aspect ratio of the mapped image.

The effect of scaling a texture map is illustrated in the following examples:

Example 1 – scale of 1 is applied to the color image



Example 2 – scale of 0.5 is applied to the color image

The material used in the above renders is a plastic material with the following properties:

- pigmentColor: (0.8, 0.7, 0.1)
- map_pigmentColor: textures\colorTexture.jpg
- scale_pigmentColor: (1.0, 1.0) or (0.5, 0.5)
- roughness: 0.0

Notice how with smaller scale value, the texture map appears to be larger.

TEXTURE MAP OFFSET

The offset of texture map translates the texture map in the U and V directions, by applying the specified shift in U and V directions

The following examples display the effect of varying the texture map offset

Example 3 – offset of 0 is applied to the color image

Example 4 – offset of 0.5 is applied to the color image

The material used in the above renders is a plastic material with the following properties:

- pigmentColor: (0.8, 0.7, 0.1)
- map_pigmentColor: textures\colorTexture.jpg
- map_bump: textures\bumpTexture.jpg
- offset_pigmentColor: (0.0, 0.0) or (0.5, 0.5)
- roughness: 0.0

GENERIC TEXTURE MAP TYPES

Some texture maps types can be applied to any material type, increasing the possibilities that can be generated from a single material type.

The common advantage of these map types is that they can be easily generated and applied instead of modeling similar effects by the user, and therefore they help save considerable modeling time by the artist, and help reduce geometric complexity.

OPACITY MAP

opacity maps give the user the ability to specify holes, transparent areas, and opaque areas on the mapped object. The map is defined as a texture map applied to MaterialMapOpacity property of any material.

The image itself contains the opacity data in the following way:

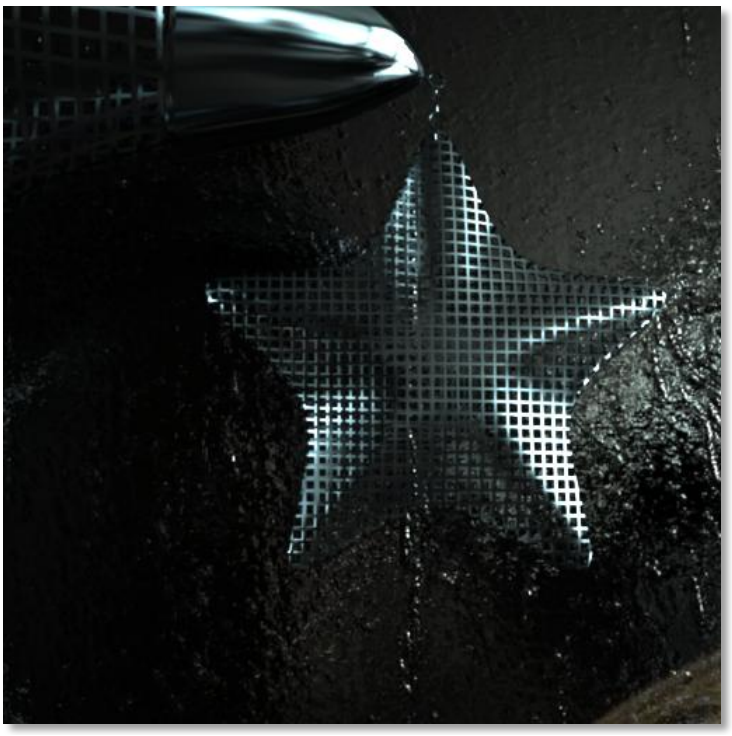
- Black pixels in the image map represent holes
- Gray pixels represent semi-transparent areas (as brightness increases, opacity increases)
- White pixels represent opaque areas

The only additional controls for opacity maps are the scale and offset, specified in MaterialScaleOpacity and MaterialOffsetOpacity respectively.

The following example shows opacity map in action, the map consists of a black grid on top of a white background.

Example 1 – opacity map: black grid over white background

The second example shows a close up on the start with an opacity map of thin white grid on top of a black background.



Example 2 – opacity map: white grid over black background

A bump map is used to simulate the look of geometric details on objects, so it helps the user avoid the inefficiencies when modeling these details in the geometry itself.

The bump map can be set to a texture of any format of the previously mentioned formats, where the user needs to set the MaterialMapBump property of any material to the desired texture. Notice that even when a colored image is supported, it is still utilized as if it is in grayscale since bump map utilizes the intensity and not the color of the pixels.

The bump map basically simulates the effect of changing the surface level of the object, where bright pixels in the bump map image simulate heightened areas, and dark pixels represent lowered areas.

The following images clearly display the effect of bump maps, and illustrate the geometric details effect that results from applying bump maps to a simple geometry.

Example 3 – bump map: wrinkled texture

The material used in the above renders is a metal material with the following properties:

- reflectance: (0.35, 0.35, 0.35)
- map_bump: textures\bumpTexture.jpg
- scale_bump: (1.0, 1.0)
- strength_bump 4
- roughness: 0.6

On top of the regular texture, scale, and offset properties, the bump map has one additional parameter, MaterialStrengthBump, which can be set to any positive value, and gives the user the ability to modify the bump effect where higher values make the effect stronger and more pronounced and therefore cause object to appear less smooth.

The following examples clearly display the effect of modifying the MaterialStrengthBump property:

Example 4 – bump strength: 0.5

Example 5 – bump strength: 1.5

Example 6 – bump strength: 2.5

Notice in the above renders how increasing the bumpiness of an object makes it appear less smooth, and how applying a bump map makes the 3d object appear as if it had more detailed geometry.

The material used in the above renders is a metallic paint material with the following properties:

- coatThickness: 0.1
- flakesSize: 0.0002
- flakesCoverage: 0.2
- flakesColor: (0.9, 0.4, 0 .9)
- map_bump: textures\bumpTexture.jpg
- strength_bump: 0.5, 1.5, or 2.5
- baseColor: (0.1098, 0.5490, 0.7941)

A normal map, similar to bump map, is used to simulate the look of geometric details on a 3d object. The difference is that in the case of bump map, a grayscale image maps the height of the surface, whereas in the normal map a colored (RGB) image is used to map the direction of the normal in addition to its height. As such, a normal map enables finer control, but requires three channels (R,G, and B) to perform normal calculations.

SimLab RT utilizes an efficient optimized implementation of normal maps, and therefore the user should not notice efficiency problems when using normal maps compared to a naive implementation.

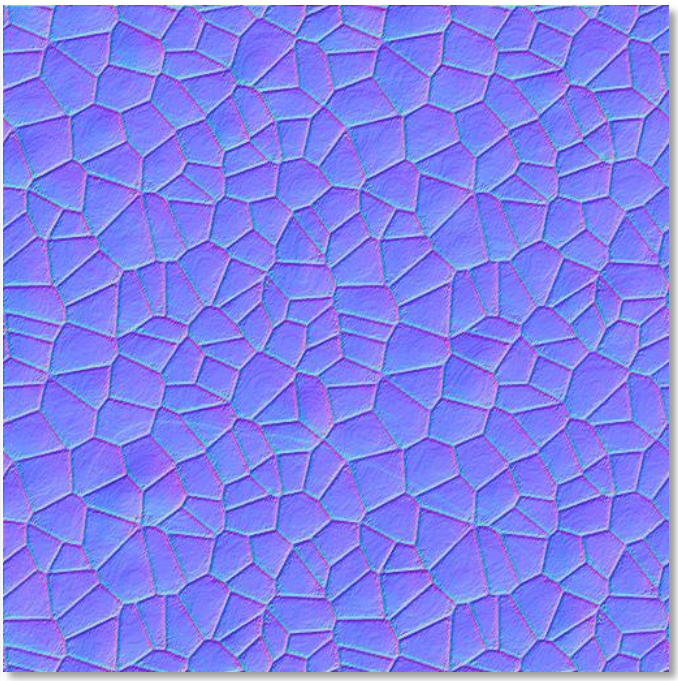
In a normal map, a blue color of (126,126,256) or (0.5f, 0.5f, 1.0f) is used to signal no change in the original normal direction of the object, any other value will change the original orientation of the normal.

Normal maps are generally generated by specialized software, which should give an RGB normal map image according to the desired effect by the user, in addition, lots of usable normal maps can be found on the net.

Another similarity to the bump map, is that the strength of the effect can be controlled using the strength_normal parameter.

The following image shows the effect of using normal maps, the used normal map itself is also displayed so that the user gets an idea of how normal maps usually look.

Example 7 – normal mapped objects



Example 8 – the normal map applied to crescent and star

The material used in the above renders is a metal material with the following properties:

- reflectance: (0.45, 0.45, 0.85)
- map_normal: textures\bumpNormal.jpg
- strength_normal: 0.85
- roughness: 0.0

The next example shows that normal maps can be applied to glass as well. As mentioned earlier, applying bump or normal maps result in decreasing the smoothness of the material, which is the same as saying that they increase the roughness of

the material. This would explain why the material below looks a lot like a rough glass even though the roughness value of the material itself was set to zero.

Example 9 – normal map applied to a smooth glass material

The material used in the above renders is a glass material with the following properties:

- reflectance: (0.45, 0.45, 0.85)
- map_normal: textures\bumpNormal.jpg
- strength_normal: 0.85
- scale_normal (0.8, 0.8)
- roughness: 0.0

LIGHTS

INTRODUCTION

SimLab RT supports common computer graphics light types, and enables the user to control various properties of these lights to create the desired lighting and mood in the rendered image.

Some light types are physically implausible, such as point lights, but are common in computer graphics and so are included in SimLab RT. Notice, however, that using physically based lighting usually creates more realistic images and so should be the user's first choice for lighting an image.

Such physically based lighting types include: Environment lights, area lights, and mesh emitters. As will be shown later, these light types create more believable images and more realistic shadows.

The goal of this section is to give a detailed description of different light types including:

- Point light
- Directional light
- Distant light
- Spot light
- Ambient light
- Environment light
- Area light
- Mesh emitters

POINT LIGHT

DESCRIPTION

Also known as 'omni light', the point light represents a light which emits in all directions uniformly. It is physically implausible since it has no volume, and thus would result in strong and harsh shadows.

The user can set the position and strength of the point light. The strength of the point light is given by a physical quantity known as the radiant intensity (measured in Watts/Sr).

SAMPLE RESULT

The sample to the right shows a rendering lit by a single point light, with white color and strength of 45. Notice the harsh shadows created by point lights. This is a side effect of the fact that point light do not have a physical volume.

PARAMETERS

Name	Type	Range	Default	Description
LightPosition	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf}	0.0f, 0.0f, 0.0f	Position of point light
LightColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f}	1.0f, 1.0f, 1.0f	Color of point light

LightStrength	float	0.0f - inf	50.0f	Strength of the point light

DIRECTIONAL LIGHT

DESCRIPTION

The directional light represents a light which emits in a single direction only, and thus, it creates sharp shadows that represent the projection of objects in the direction of the light.

The user can set the direction, color, and strength of the directional light, whereas the directional light has no position. The color multiplied by strength gives a physical quantity known as the irradiance of the light (measured in Watts/m^2).

SAMPLE RESULT

The sample below shows a rendering lit by a single directional light, with white color and strength of 5. The direction of the light is set so that it appears to be coming from above and slightly to the right of the camera. Notice the harsh shadows created by directional lights. This is a side effect of the fact that directional lights emit in a single direction only.

PARAMETERS

Name	Type	Range	Default	Description
LightDirection	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf}	NA	Direction of emission of the light
LightColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f}	1.0f, 1.0f, 1.0f	Color of directional light
LightStrength	float	0.0f - inf	5.0f	Strength of the directional light

DISTANT LIGHT

DESCRIPTION

Distant light is used to simulate a very far light source which emits light in a cone of directions, this makes it especially useful for simulating distant light sources such as the sun.

As such, there are common properties that are shared with directional light, namely: the direction of emission, color, and strength. In addition, the user can set another parameter to control the spread of the cone of directions in which light rays are emitted. The color multiplied by strength gives a physical quantity known as the radiance of the light (measured in Watts/m^2.sr).

SAMPLE RESULT

The sample to the right shows a rendering lit by a single distant light, with white color and strength of 1. The direction of the light is set so that it appears to be coming from above and slightly to the right of the camera.

Notice the smooth shadows generated by distant lights, this is a result of the fact that a point at a specific position does not receive light coming from a single direction only, but rather each point now receives light coming from a cone of directions

PARAMETERS

Name	Type	Range	Default	Description
LightDirection	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf}	NA	Direction of emission of the light
LightColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f	1.0f, 1.0f, 1.0f	Color of distant light

		}		
LightStrength	float	0.0f - inf	1.0f	Strength of the distant light
LightHalfAngle	float	0.0f - 45.0f	5.0f	Half of the spread angle of the light (in degrees)

EXAMPLES

The following examples show how increasing the halfAngle value results in less defined (ie more blurry) shadows and high-lights. In the following, halfAngle values is set to 15 instead of 5 as in previous example:

Example 1 – halfAngle: 5

Example 2 – halfAngle: 15

SPOT LIGHT

DESCRIPTION

A spot light emits light in a cone of directions starting from a fixed position. In addition, spot light has a falloff region, where the light emits with maximum power in a smaller cone of directions, and then light power starts decreasing gradually until no emission occurs beyond the larger cone of directions.

The user has the ability to set the position, direction, color, strength, minimum angle (corresponding to the angle of the smaller cone in which the light emits with full power) and maximum angle, where the light starts the falloff in power at minimum angle and ends with no emission at maximum angle.

SAMPLE RESULT

The sample to the right shows a rendering lit by a single spot light, with white color and strength of 15, minimum angle is 10 degrees, and maximum angle is 25 degrees.

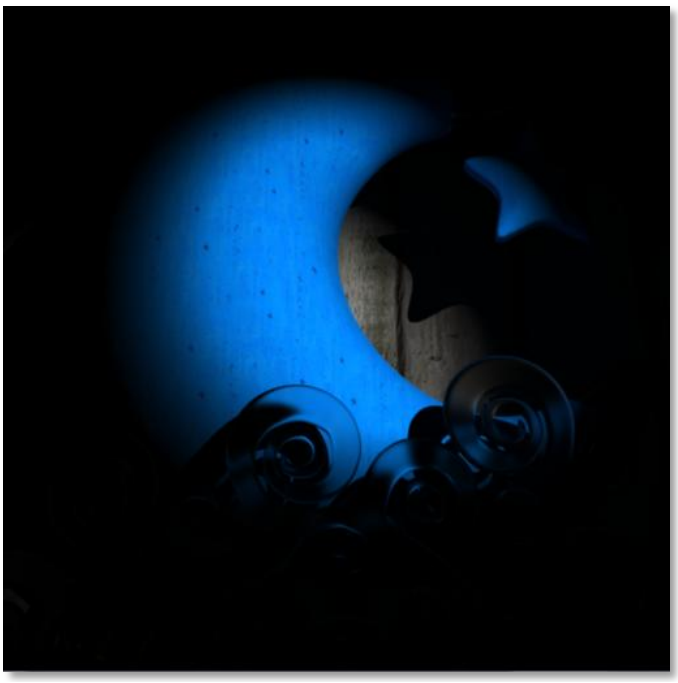
Notice the sharp shadows when a spot light is used, due to the fact that (similar to point light) each specific position in the scene is hit by light from one possible direction only.

PARAMETERS

Name	Type	Range	Default	Description

LightPosition	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf }	NA	Position of spot of the light
LightDirection	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf }	NA	Direction of emission of the light
LightColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f, 1.0f, 1.0f	Color of spot light
LightStrength	float	0.0f - inf	1.0f	Strength of the spot light
LightAngleMin	float	0.0f - 45.0f	5.0f	Angle at which spot light starts falling off
LightAngleMax	float	LightAngleMin to 45.0f	25.0f	Angle after which spot light does not emit any light

Following examples illustrates the effect of maximum angle on the result. It is clear that light falls off more rapidly when max angle is decreased, and let area gets smaller as well.



Example 1 – LightAngleMax: 15

Example 2 – LightAngleMax: 25

AMBIENT LIGHT

DESCRIPTION

An ambient light gives a similar effect to lighting the scene with a uniform (single color) environment light. This means that another way to give a similar result is to set the image of the environment light to a single color image.

As a result, the image created with ambient light only has diffused, uninteresting shadows. Therefore, ambient light is usually not used by itself, but in addition to other type of lighting to create more realistic shadows. The user has the ability to set the color and strength of the ambient light.

SAMPLE RESULT

The sample to the right shows a rendering lit by ambient light only, with white color and strength of 0.8.

As clear in this example, both lighting and shadows are diffused as a result of ambient lighting.

PARAMETERS

Name	Type	Range	Default	Description
LightColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	0.5f, 0.5f, 0.5f	Color of ambient light
LightStrength	float	0.0f - inf	1.0f	Strength of the ambient light

ENVIRONMENT LIGHT

DESCRIPTION

Environment light, also called infinite light or HDRI light, is a common type of light for exterior scenes, where the light is generated from an image mapped to a sphere surrounding every point in the 3D scene.

The user has the ability to control the strength of the environment light, in addition to a local to world transformation matrix in order to rotate the environment light in any desired orientation.

SAMPLE RESULT

The sample to the right shows a rendering lit by an environment light, with strength of 1 and default orientation.

Notice how both lighting and shadows look realistic in the image.

PARAMETERS

Name	Type	Range	Default	Description
LightLocal2World	AffineSpace		Identity Affine-Space	Local to world transformation to determine the orientation of the environment light
LightL	(float,float,float)	{0.0f – inf, 0.0f – inf, 0.0f – inf }	1.0f, 1.0f, 1.0f	R,G, and B strength of the environment light
LightImage	Image file			Image file (usually a high dynamic range image such as an EXR or HDR) to be used as the source of environment lighting
LightHemispherical	bool		FALSE	when turned on, the lower hemisphere of the environment map will not emit light. This results in faster rendering in case light will not reach the objects from below, e.g. if there is a horizontal plan below the objects the prevents light from below to reach the objects

Following example illustrates the effect of setting a different transformation matrix than the identity. The following AffineSpace was applied in this case:

AffineSpace(LinearSpace3f(Vec3f(0,1,0),Vec3f(-1,0,0),Vec3f(0,0,1)),Vec3f(0,0,0))

Which corresponds to a rotation of 90 degrees about the z-axis. Notice that the position of the transformation (set to 0,0,0 in the example) does not affect the end result, since only orientation is considered in the environment light.

Example 3 – LocalToWorld is set to none-identity matrix

SKY LIGHT

DESCRIPTION

Sky light is used to simulate a physical sky with sun at different sun angles (called elevation angle), where sun angle determines the time of the day. The model is physically based, and has the ability to simulate the effect of ground albedo (ground color reflection) which in turn affects the color and feel of the sky, and can be used to exaggerate the feeling of the sky and create fantasy skies.

The user has multiple options to control the sky, including specifying the turbidity (think of it as clarity) of the sky, user can also modify the color for ground albedo, the solar elevation angle (which in turn determines the time of the day), the horizontal solar angle (acting as a rotation of the sun around the zennith), in addition to controls for changing the sky strength, and the sun strength.

SAMPLE RESULT

The sample to the right shows a rendering lit by a sky light with the following settings:

- *Turbidity = 3*
- *Ground Albedo = Gray (0.3, 0.3, 0.3)*
- *Sun elevation Angle = 18 degrees*
- *Sun horizontal angle = 65 degrees*
- *Sky strength = 0.75*
- *Sun strength = 0.75*

Notice how both lighting and shadows look realistic in the image, and match the observed day lighting effect in real life.

PARAMETERS

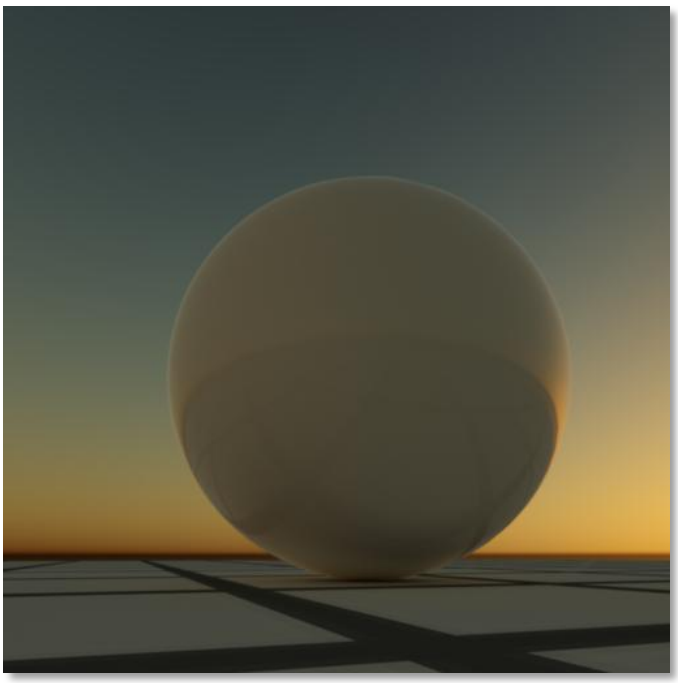
(Parameters in bold have samples in the examples section below)

Name	Type	Range	Default	Description
LightLocal2World	AffineSpace		Identity AffineSpace	Local to world transformation to determine the orientation of the sky light
LightTurbidity	float	0.0f, 1.0f, 2.0f, 3.0f to 10.0f	3.0f	Turbidity of the sky, which determines ratio of large particles that affect scattering of the light in the sky. Low values present a clear sky, while high values make the sky appear more hazy
LightSkyStrength	float	0.0f - inf	1.0f	The strength of the sky part of the sky model

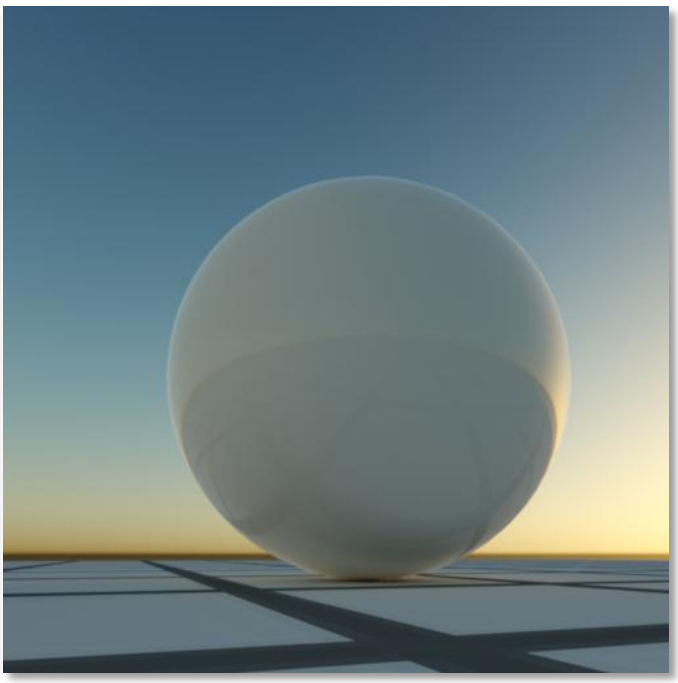
LightSunStrength	float	0.0f - inf	1.0f	The strength of the sun part of the sky model
LightAlbedo	(float,float,float)	{0.0f – inf, 0.0f – inf, 0.0f – inf }	0.3f, 0.3f, 0.3f	Ground albedo that affects the scattering, and hence the tone of the sky. Check the examples below to observer the effect of the ground albedo
LightSolarElevation	float	0.0f - 90.0f	18.0f	Angle which the sun makes with the horizon, where zero degrees means the sun is at the level of the ground, and 90 degrees means the sun is in the middle of the sky
solarHorizontalAngle	float	0.0f - 360.0f	0.0f	rotation angle to rotate the sun around the zenith without affecting the sun elevation
renderSolarDisc	bool		TRUE	Option to render the solar disc in the sky or not, when set to fales, the solar disc is not rendered in the sky, but the effect of the sun light is still there

EXAMPLES

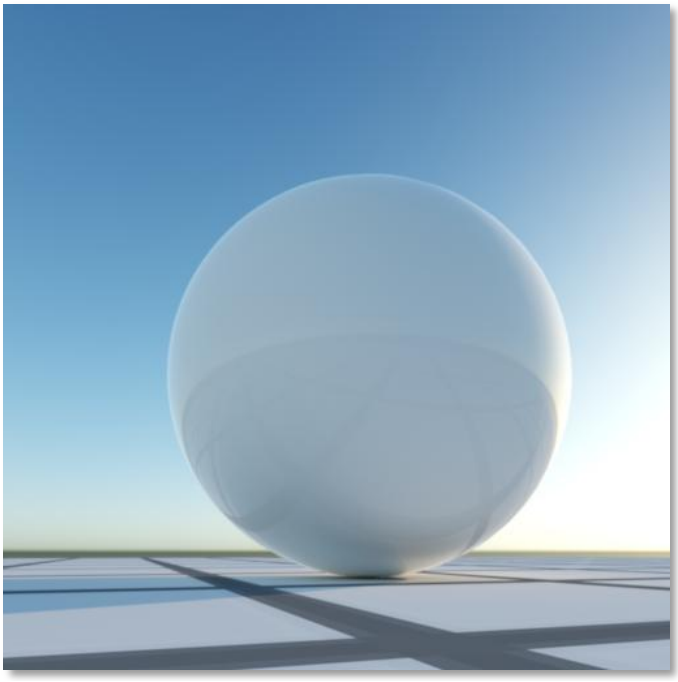
The first set of examples displays the effect of LightSolarElevation variable on the sky light system. Notice the solar elevation angle (in degrees) determine the time of the day, with low angles representing sunrise or sunset, and high angles bring the sun towards the middle of the day.



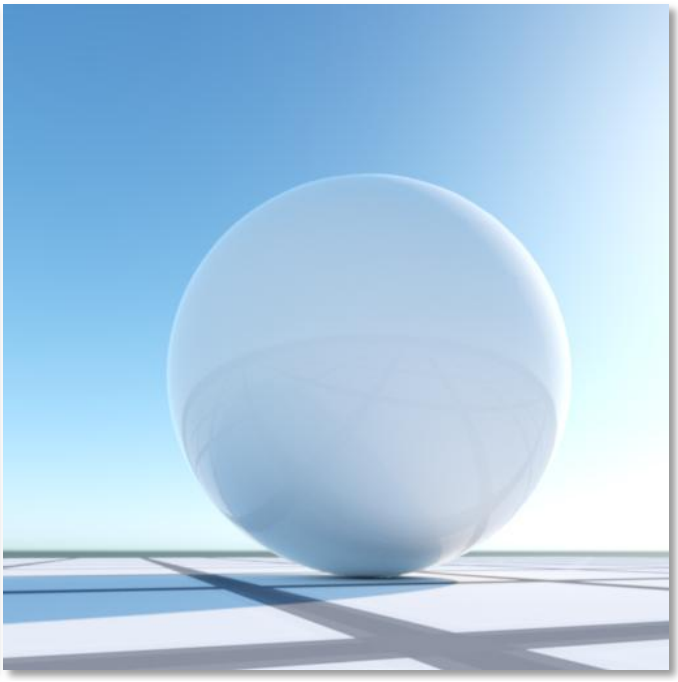
Example 4 – LightSolarElevation : 1.0



Example 5 – LightSolarElevation : 5.0



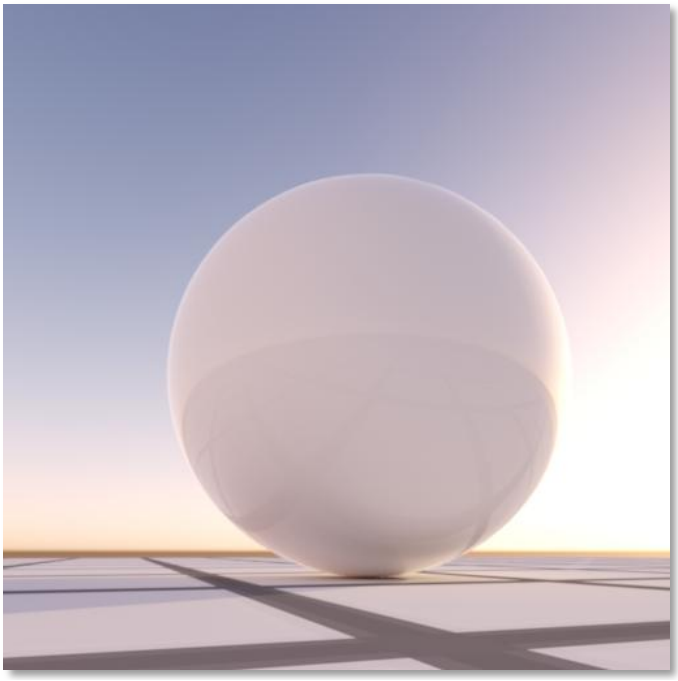
Example 6 – LightSolarElevation : 15.0



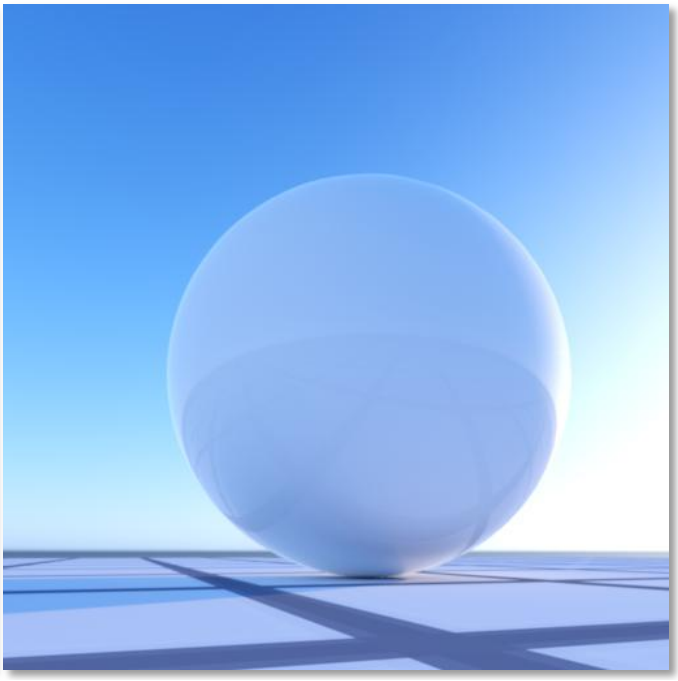
Example 7 – LightSolarElevation : 33.0

Example 8 – LightSolarElevation : 75.0

The following examples display the effect of ground albedo to the final look of the sky, notice how a reddish ground albedo modifies the overall tones of the sky environment in a subtle, but noticeable way



Example 9 – Ground Albedo : Reddish(0.6, 0.35, 0.3)



Example 10 – Ground Albedo : Bluish(0.3, 0.35, 0.6)

EMITTER LIGHT

DESCRIPTION

An emitter light is a result of applying a mesh emitter material to any of the object in the scene. Therefore, mesh emitter material is discussed from the other perspective (i.e. the materials perspective) in the materials section.

Since an emitter light is a physically plausible light type (with light volume larger than zero, and light emission in all directions from emitter object), this light type results in interesting and realistic soft shadows and lighting of the scene.

Emitter lights are sampled very efficiently in SimLab RT, so the user should not feel constrained at all to use other light types instead of emitter light for efficiency reasons.

Similar to other light types, the user has the ability to set the color and strength of the emitter light.

SAMPLE RESULT

The sample to the right shows a rendering lit by a spherical emitter light only, with purple color (0.3, 0.2, 0.8) and strength of 6. The mesh emitter sphere is placed behind the camera and to its right.

Both soft shadows and interesting lighting are clear in this example

PARAMETERS

Notice that since a mesh emitter starts as a material applied to the object, the properties below are actually material properties, and are the same as the ones listed under mesh emitter material parameters, included here for reference only.

Name	Type	Range	Default	Description
MaterialC	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	1.0f, 1.0f, 1.0f	Color of the mesh emitter material, controls the light color emitted from the object
MaterialStrength	float	0.0f - 1.0f	0.0f	The strength of the mesh emitter light, the higher the strength the stronger the light source

CAMERAS

INTRODUCTION

In 3D computer graphics, a camera controls the view from which the user observes the 3D scene.

SimLab RT cameras are capable of rendering 3D depth of field effect by setting the focal distance and lens radius as we will discuss below. In addition, SimLab RT also has the unique ability to render multiple cameras in the scene at the same time, each with its own view and depth of field parameters, resulting in a rich render with multiple views in the least time, without any required post-processing effort. This later feature is given the name "MultiCamera render" in SimLab RT's terminology, and is discussed in its own section below.

SUPPORTED CAMERA TYPES

As of now, SimLab RT supports perspective cameras only. A perspective camera has a distinct feature called "foreshortening", which causes near objects to appear larger than distant ones, and parallel lines to lose their parallelism from camera perspective.

Notice that the human eye can be considered a perspective camera since foreshortening occurs naturally in human vision. A later section will discuss perspective camera parameters in detail.

COMMON CAMERA PROPERTIES

Name	Type	Range	Default	Description
CameraUp	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	NA	The up direction of camera, needed to remove ambiguity about the up direction of the camera when only viewing direction is

				known
CameraLookAt	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf }	NA	Position of the point at which camera is looking
CameraPos	(float,float,float)	{-inf to inf, -inf to inf, -inf to inf }	NA	Position of camera itself
CameraAspectRatio	float	0.0f - inf	1.0f	Ratio of scale in the y-direction divided by scale in the x-direction of the raster image
CameraLensRadius	float	0.0f - inf	0.0f	The radius of the lens used in depth of field calculations. As lens radius is increased, the image gets more blury, and objects closer to the focalDistance get more defocused
CameraFocalDistance	float	0.0f - inf	Distance between CameraPos and CameraLookAt	Distance between camera and its focal plane, which appears in focus in the rendered image

CAMERA DEPTH OF FIELD

As shown in the common parameters of cameras, there are two parameters which control the effect of 3D depth of field in the rendered image: lensRadius and focalDistance.

The lensRadius controls the amount of defocus in objects that don't lie on the focal plane. As this value is increased, the image looks more blurred, and more objects get defocused.

The following renders illustrate the effect of increasing lensRadius while keeping focalDistance at a constant value. The focalDistance is picked at a point on the front glass (cloud) object, and notice how increasing the lensRadius caused the moon and the overall image to appear more blurry than first image.

Example 1 – LensRadius : 1

Example 2 – LensRadius : 2

The other parameter, focalDistance, controls the distance from camera position to the focal plane, which is the plane that appears in focus, this value can be picked based on which objects the user wants to appear focused, and usually that's the most important object in the render from user's perspective.

The following examples display the effect of changing the focalDistance. The focal distance in the first render was picked on the star, whereas in the second render it was picked on the cloud. Notice how this results in changing the object of interest among the two renders of the same scene.

Example 1 – Focal Distance set to the star

Example 2 – Focal Distance set to the cloud

PERSPECTIVE CAMERA PROPERTIES

In perspective camera, in addition to common camera parameters, the user can manipulate the following parameter:

Name	Type	Range	Default	Description
CameraAngle	float	0.0f - 180.0f	64.0f	Field of view (FOV) angle of the camera, in degrees

The FOV angle (in degrees) controls the amount of foreshortening in the render. As FOV angle increases, the camera is able to view objects at a wider angle, and as a result, distant objects appear smaller as FOV is increased.

In real life, the field of view angle of a lens is related to its focal length.

MULTICAMERA FEATURE

SimLab RT introduces an innovative feature to render multiple cameras (each with its own parameters, such as view direction and depth of field parameters) and render them on the same raster image at the same time, without any loss of rendering effeciency. The user controls the region (in the raster image) and the parameters of each of the cameras.

The sample image to the right is rendered with SimLab RT's multicamera feature, meaning there was no need for the user to render multiple images (or even guessing about the final look) and then arranging the renders in another 2D application.

SimLab RT saves time and effort, and displays the final look in just few seconds.

BACKGROUND

INTRODUCTION

In order to help the user get a final image directly out of SimLab RT, the user has multiple options to display an image as the background of the rendering.

The options currently supported for a background are:

1- Spherical background
2- Planar backplate
3- Environment map

The order of the list above is important, so for example if SimLab RT is provided with both a spherical background and a planar backplate, SimLab RT will use the spherical background and ignore the backplate, etc. The following sections detail the different options available to the user and give examples of each.

It is worth noting that in case the environment map gives the desired result, it is preferred to be used over spherical background for slight rendering performance gain.

SPHERICAL BACKGROUND

USAGE

The spherical background enables the user to specify an image to be applied to a background sphere object. It includes options to rotate the image or resize the sphere. In addition, there is a control to apply the mapped image to a hemi-sphere instead of a full sphere, which is useful in case the user prefers a flattened ground.

PARAMETERS

Name	Type	Range	Default	Description
SphericalLocal2World	AffineSpace		Identity AffineSpace	Local to world transformation to determine the orientation of the spherical background
SphericalL	(float,float,float)	{0.0f – inf, 0.0f – inf, 0.0f – inf }	1.0f, 1.0f, 1.0f	R,G, and B strength of the spherical background
SphericalImage	Image file			Image file to be used as the spherical background
SphericalRadius	float	0.0f - inf	1000.0f	Radius of the sphere to which the image is applied
SphericalHemispherical	bool		FALSE	When enabled, the lower half of the spherical image will be mapped to a disc, so the full image is applied to a hemisphere instead of a full sphere

EXAMPLES

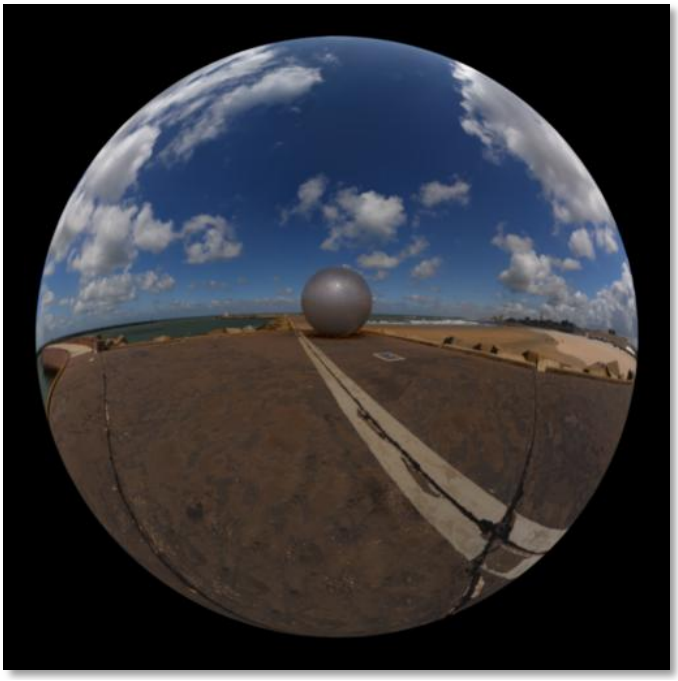
The samples below show how the parameters above affect the spherical background.

The first sample shows the difference between disabling and enabling the hemispherical option, notice how the ground is flattened in the second case. In both cases, the radius was set to 750.

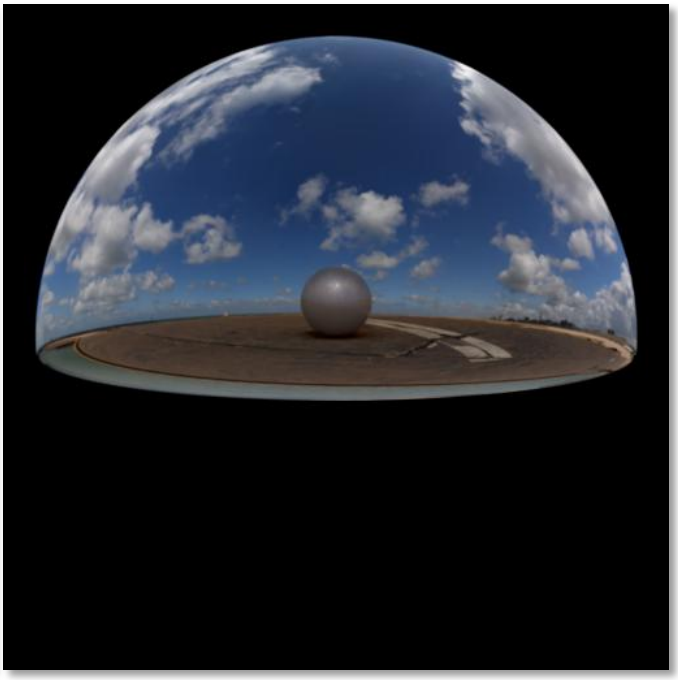
Example 1 – Hemispherical set to false

Example 2 – Hemispherical set to true

To further illustrate the difference between the two options, the images below display the difference from a distant point of view



Example 3 – Hemispherical set to false



Example 4 – Hemispherical set to true

The following set of examples display the result of varying the spherical radius, notice how a smaller portion of the image appears within the field of view when we increase the spherical radius.

Example 5 – Radius set to 500

Example 6 – Radius set to 1500

PLANAR BACKPLATE

USAGE

The planar backplate is used to apply an image as a background behind all 3D objects. It is commonly used in computer graphics specially with a matching HDR environment image to create a consistent feel of the image.

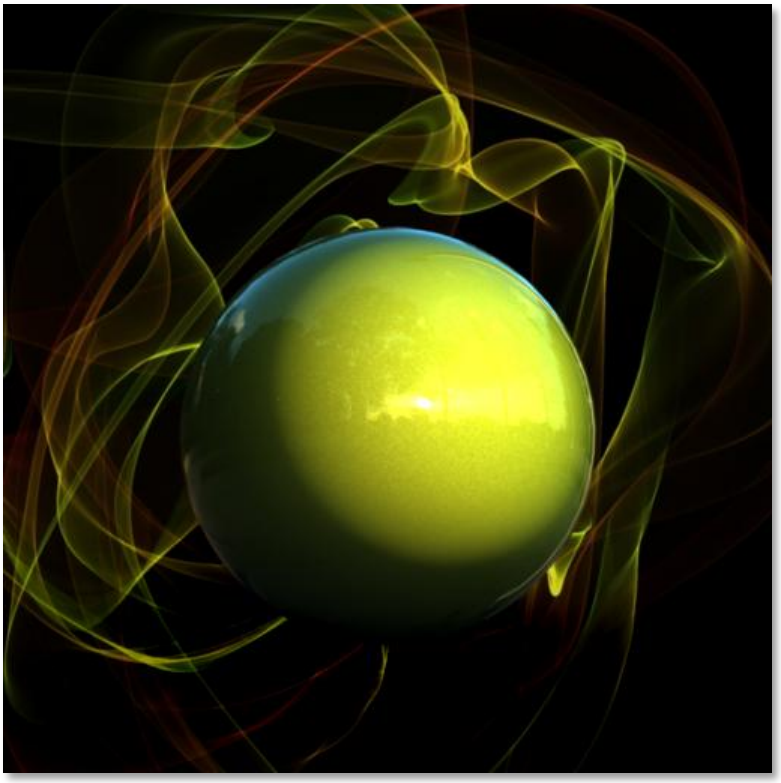
PARAMETERS

Name	Type	Range	Default	Description
RendererBackplate	Image file			Image file to be used as the planar backplate

EXAMPLES

The sample image to the right displays a sample of utilizing a planar backplate in your renders.

A planar backplate along with a matching HDRI environment lighting is a powerful combination to produce realistic results



ENVIRONMENT MAP

USAGE

The environment map is displayed in the background only if no spherical background or planar backplate were specified

Environment map is also utilized as a lighting source, that's why all parameters and examples of the environment map are discussed under the relevant section in the light sources chapter.

GROUND OPTIONS

INTRODUCTION

Instead of forcing the user to model a ground object, SimLab RT provides an infinite ground plane with multiple options. These options help the user quickly get to his desired composition with least effort.

The user can control both ground shadows and ground reflections, along with multiple settings of each to be chosen by the user to create the best effect, the sections below detail these options along with some examples.

GROUND SHADOWS

USAGE

Ground shadows are used to make the object feel to be in contact with the ground, it is also very useful when using a background image (see section above) to make the object appear blended with the background.

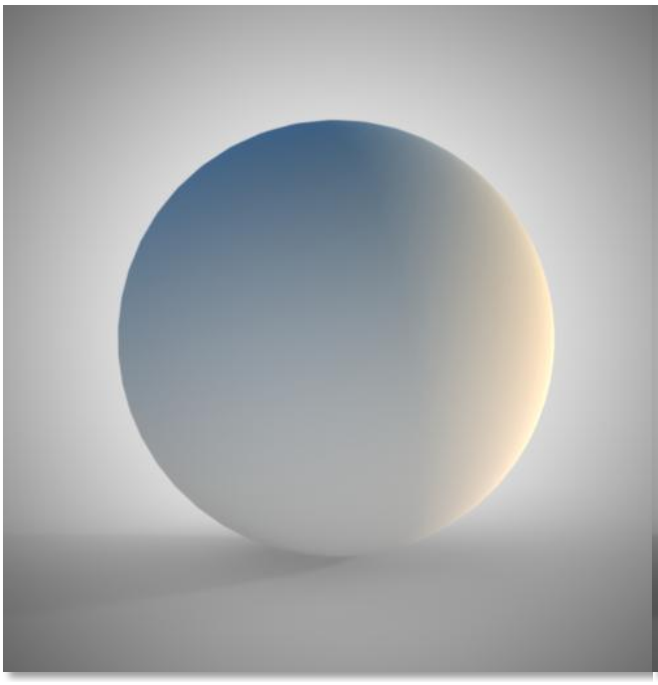
The user has multiple controls to select desired artistic properties of the ground shadow, including its strength and color as will be discussed in the following section.

PARAMETERS

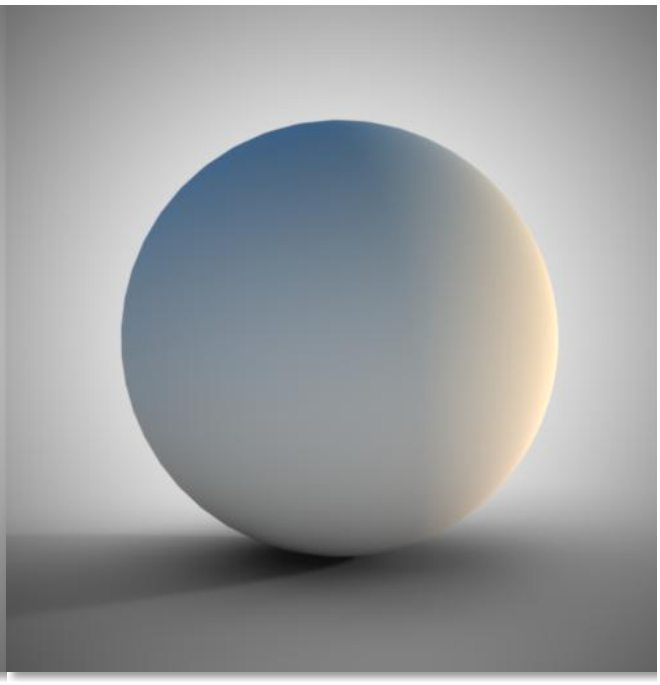
Name	Type	Range	Default	Description
GroundShadowColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	(0.5f, 0.5f, 0.5f)	Color of the ground shadow, default color is black, where white color results in no visible shadows at all
GroundShadowStrength	float	0.0f - 10.0f	0.0f	The strength of the ground shadow, where higher values result in more pronounced shadows. Strength value of zero disables the ground shadow.

EXAMPLES

The first set of examples for ground shadow options displays the effect of varying the ground shadow strength, notice how increasing the strength results in darker and more pronounced ground shadows

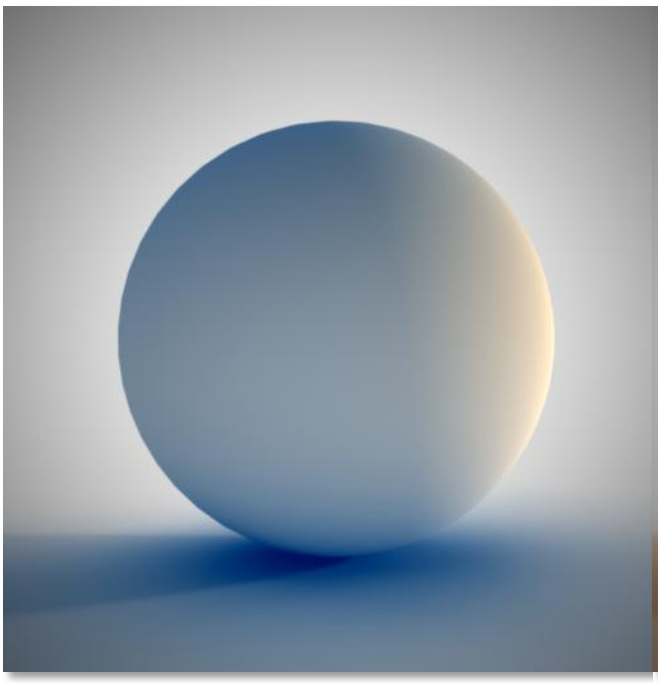


Example 1 – Shadow strength set to 1

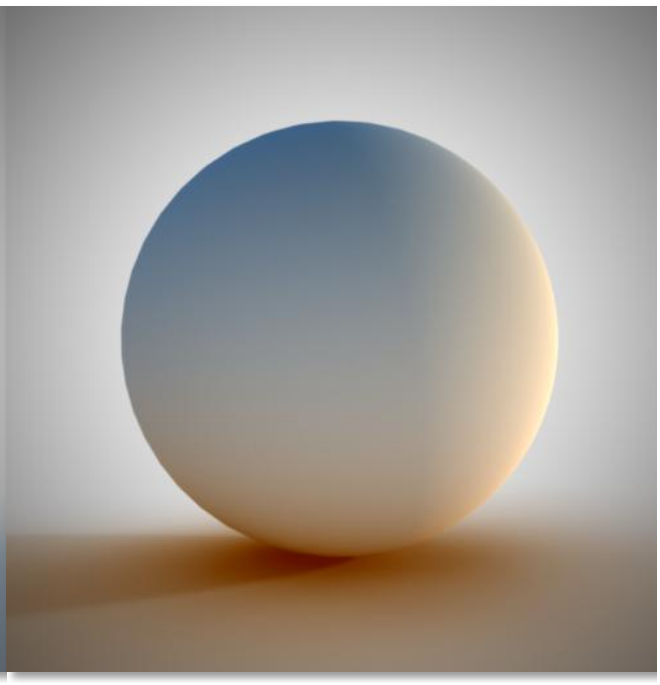


Example 2 – Shadow strength set to 2

The next set displays the effect of varying shadow color, notice how it affects the mood of the rendered image



Example 3 – Shadow Color set to blue - (0.3, 0.5, 0.7)



Example 4 – Shadow Color set to brown - (0.7, 0.5, 0.3)

GROUND REFLECTIONS

USAGE

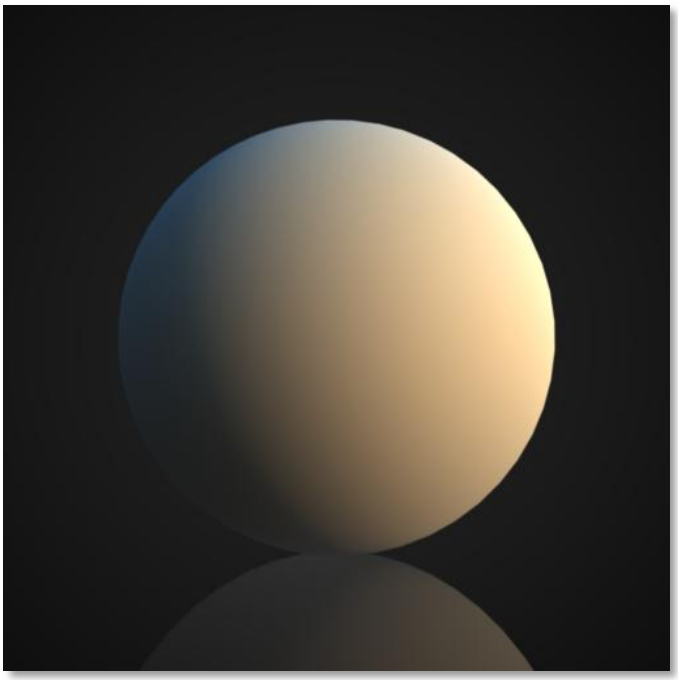
Ground reflections option turns the ground into a reflective plane. Similar to ground shadows; the user controls reflection strength and reflection color. Besides these two settings, the user has options to control the glossiness (or roughness) of the reflections, and whether to reflect the environment map in addition to reflecting 3d objects of the scene.

PARAMETERS

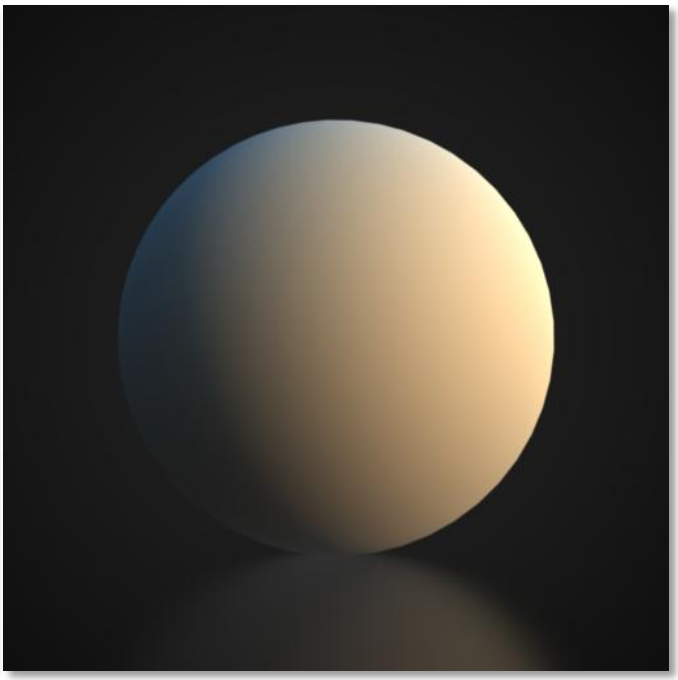
Name	Type	Range	Default	Description
GroundReflectionColor	(float,float,float)	{0.0f – 1.0f, 0.0f – 1.0f, 0.0f – 1.0f }	(0.5f, 0.5f, 0.5f)	Color of the ground reflection: default color is grey, where black color results in no reflections at all
GroundReflectionStrength	float	0.0f - 10.0f	0.0f	The strength of the ground reflection, where higher values result in stronger reflections. Strength value of zero disables the ground reflection
GroundReflectionEnableFresnel	bool	true or false	FALSE	Determines whether Fresnel reflections are enabled or disabled. Fresnel reflections can be disabled to simulate metallic looking reflections
GroundReflectionRoughness	float	0.0f - 1.0f	0.01f	Roughness of the ground reflection, higher value increases glossiness of the reflection
GroundReflectsEnvironment	bool	true or false	FALSE	Determines whether to reflect the environment map or not, this option is useful in some cases to make the ground blend better with the image

EXAMPLES

First set of examples of ground reflection display the effect of reflection roughness, notice how increasing the roughness values results in more blurry reflections

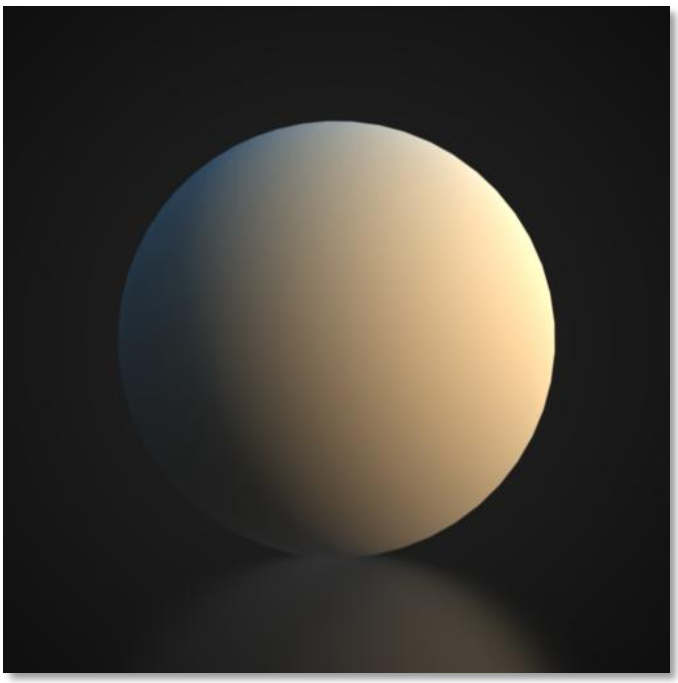


Example 5 – Ground reflection roughness set to 0.0

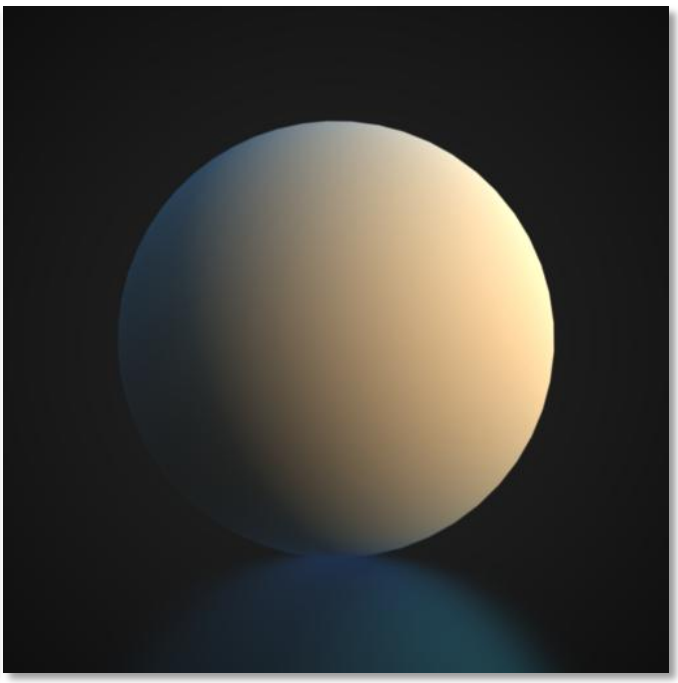


Example 6 – Ground reflection roughness set to 0.4

The following two examples display the effect of reflection color

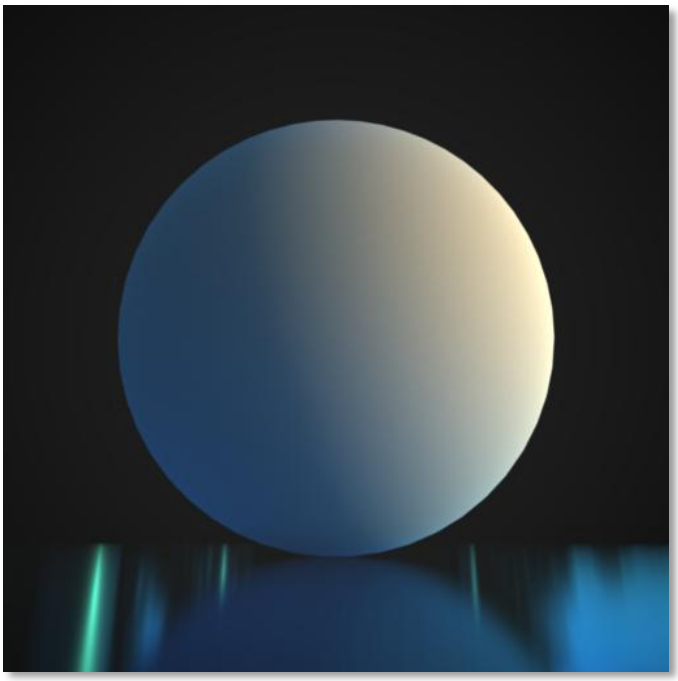


Example 7 – Ground reflection color left as grey



Example 6 – Ground reflection color set to blue(0.1, 0.5, 0.8)

And the last example in this section clarifies what happens when the 'reflect environment' option is enabled. Notice how the ground in this case reflects the HDR environment in addition to reflecting 3d objects



Example 7 – environment reflection option is enabled

TONEMAPPING

INTRODUCTION

SimLab RT supports tone mapping to provide different approximation techniques for viewing the internal HDR image representation, and to display high dynamic range images into common 32 bit computer displays while retaining the fidelity of the original HDR image.

Currently, SimLab RT supports multiple global tone mapping techniques, each with its own set of parameters as detailed in the following section.

SUPPORTED TONE MAPPING TYPES

As of today, SimLab RT supports 2 different tone mapping techniques:

- Uncharted2 (Default tone mapping technique): a powerful tonemapping technique, which retains contrast of the rendered image, and has a control for output exposure.

- Reinhard: a handy and basic tonemapper that works fine in most cases, and in general results in low contrast rendered image.

DEFAULT TECHNIQUE (UNCHARTED 2)

This tone mapper is based on the approach used in the PS3™ game Uncharted 2™, and results in a very efficient and powerful tonemapping technique.

PARAMETERS

Name	Type	Range	Default	Description
TonemapperGamma	float	0.0f - 5.0f	1.0f	Value to apply as gamma correction for the output
TonemapperVignetting	bool		FALSE	Enable or disable vignetting effect
TonemapperExposureBias	float	0.0f - 5.0f	2.0f	Exposure adjustment for the tonemapper

TM1 (REINHARD TECHNIQUE)

There are many different approaches for Reinhard tonemapping, but in this release of SimLab RT, support for the simplest Reinhard operator was added: Color = Color/(1+Color)

PARAMETERS

Following is the set of parameters that can be changed in Reihnhard tone mapping:

Name	Type	Range	Default	Description

TonemapperGamma	float	0.0f - 5.0f	1.0f	Value to apply as gamma correction for the out-put
TonemapperVignetting	bool		FALSE	Enable or disable vignetting effect