

Reference Manual

Reference Manual

Copyright © 1985 ... 2008 Realsoft Graphics Oy

All rights reserved. This documentation contains proprietary confidential information and trade secrets of Realsoft Graphics Oy, Realsoft Oy and Realsoft Graphics Oy.

THIS SOFTWARE AND RELATED MATERIAL IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. IN NO EVENT SHALL REALSOFT GRAPHICS OY BE LIABLE FOR ANY DAMAGES CAUSED USE OF THIS MATERIAL.

Table of Contents

1. Realsoft 3D	1
Models	1
Channels	1
Choreography Classes	2
File Rendering Configurations	5
Geometric Objects	9
Grids	85
Images	86
Materials	86
Output Objects	119
Post Particle Effects	124
Post Image Effects	127
Rendering Settings	138
View Models	145
Animation Object	145
Startup options	146
Scripting	148
Built-in JavaScript Classes	148
Tools	174
3D painting Tools	174
Analytic Tools	179
Boolean Tools	184
Building Construction Tools	185
Construction Tools	188
Creation Tools	196
Deformation Tools	200
Foot Print Tools	206
Geometry Manager Tools	207
View Control Tools	208
Lattice Mapping Tools	212
Light Source Creation Tools	216
Material Mapping Tools	218
Miscellaneous Tools	221
NURBS Curves and Surfaces	226
Plant Tools	244
Subdivision Surface Tools	245
Simulation Tools	262
Transformation Tools	263
Triset Tools	267
Unified Creation Tools	268
User Interface	277
Geometry Managers	277
Macro GUI gadgets	278
Windows	278
Windows	282
Advanced Selection Window	282
Animation Settings	283
Animation Window	284
Attributes Window	286
Choreography Window	286
Color Window	291
Curve Editor	292
File Rendering Window	294
Info Window	295

Material Browse Window	295
Numeric Window	295
Object Browse Window	296
Options Window	296
Property Window	298
Scripting Window	299
Select Window	299
Status Window	313
Tool Bar	313
Tool Control Bar	314
UV Editor	314
UV Editor Properties	316
View	316
View Control Bar	324
View Properties	327
2. BatchRen	335
Description	335
Startup Options	335
Examples	336
3. BatchRenW	337
Description	337
Using BatchRenW	337
4. RendD	338
Description	338
Startup Switches	338
Examples	340
5. RendW	342
Description	342
Pull-down Menus	342
Startup Options	342
The Control Window	343
6. ShowImg	346
Pull-down Menus	346
Startup Parameters	346
7. StartOpts	348
Description	348
Startup options	348
8. RsView	350
Description	350
Pull-down Menus	350
Startup Parameters	350
Index	352



Chapter 1. Realsoft 3D

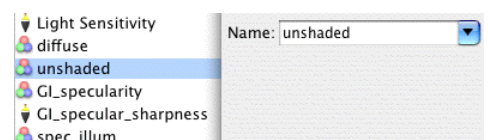
Models

Channels

Channel objects are very simple. They only specify a name and a the amount of floating point valued sub channels. However, channels play an important role when building customized rendering styles and solutions. By using channels, you can attach an unlimited amount of attributes, which are available in material shaders and rendering engine modules, to complete objects as well as their geometry points.

Currently three types of channels are available:

- **Float**: a single floating point value channel
- **Color**: a RGB triplet of floating point values
- **Vector**: three values for the x, y and z dimensions

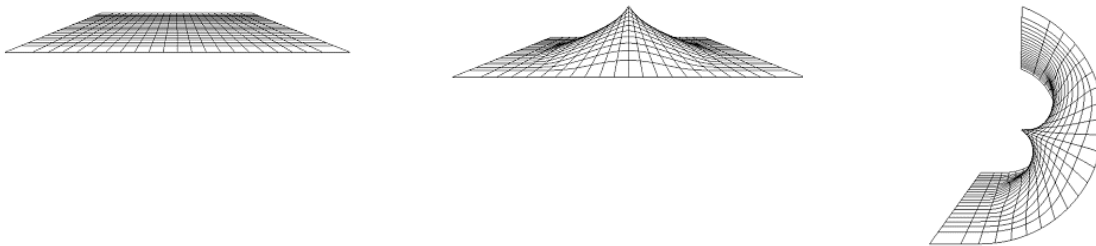


Choreography Classes

Realsoft 3D choreography system is based on *construction stack* idea. A number of choreographs can be attached to an object. Choreographs are executed one by one and each choreography object modifies the state resulted by the previous choreography.

For example, a radial deformer can be used for generating a bump to a mesh. Then a 'bend' deformer can bend the mesh. Because choreographs modify the state resulted by the previous choreography, also the bump get bent.

When any of the choreography parameters is modified, the object is rebuild accordingly.



The first choreography defines the initial shape of the mesh. The second choreography (radial deformer) generates a vertical bump to the mesh. The third choreography bends the mesh and the bump with it.

There are two ways in which the user can modify choreography attributes:

- by modifying the actual target object, in which case the *current choreography* attempts to update its attributes accordingly. Not all choreographs support this, however. For example, you cannot modify noise parameters by editing the ripples of the target object. Keyframe choreography does support this method, by editing its animation curves according to the users actions (this process is called key frame creation).
- by editing the choreography attributes through the choreography window.

Assign

This choreography object is typically associated with the root level choreography and named 'init'. Its task is to maintain the initial state of the animated attributes.

To modify the initial values, make the init (the first) choreography the current choreography, and then modify the target object attribute.



Pose

Pose choreograph is similar to Assign choreography in that its job is to keep track of the current values of attributes. However, whereas Assign choreography saves the absolute values of attributes, Pose choreography stores attributes in relative way. Pose choreography allows you to 'translate' attribute values.

You can update the current Pose values simply by making the pose the current choreography and then modifying the attribute in question.

Pose choreography is special in that it is automatically created by the system. If animation recording is off and you modify an attribute that is already controlled by a choreography that doesn't allow target editing, a pose choreography is created to represent the attribute changes.

Keyframe

Keyframe choreography is also special in that it is automatically created by the system. If animation recording is on and you modify an attribute, key framer is created to record the attribute values in time sensitive way.

Keyframer does this maintaining animation curves. Every time you modify the target object, the keyframer inserts new point to the corresponding curve (inserts new key frames). Correspondingly, when time is changed, the choreography evaluates (interpolates) its animation curves and assigns the values back to the target object.

Noise

Noise choreography applies 3D fractal noise to the target attributes. Noise differs from assign, keyframe and pose choreographs in that its cannot update any of its parameters when the target object is modified. Noise choreography is handy for adding some randomness to animations.

Script

Script choreography can be used for controlling animated attributes via scripting languages, such as JavaScript.

Scrip choreography defines the following variables for this:

```
P - target attribute (in target object's local space)
Time - current time (or the current input attribute)
Input - input object, or null if input is time
Target - target object (whose attribute 'P' is).
```

Because target attributes are all referred through the variable 'P', scripts created this way are reusable.

The type of the 'P' variable corresponds to the type of the target attribute (integer, number, vector, vector4, axis, coordsys).

The following examples assume basic knowledge about JavaScript.

Example 1 - a constraint:

Let us imagine you have animated the 'Translate' attribute and want to write a script that constraints the value of the attribute's y component above the ground level. You can achieve this by attaching the following script choreography to the Translate attribute.

```
if(P.y < 0)
    P.y = 0;
```

Example 2 - a circular path animation:

To make an object to move along a circular path in its x-y plane:

```
// uses JavaScript Math object to evaluate a circle
P.x += Math.sin(Time * 2 * PI);
P.y += Math.cos(Time * 2 * PI);
```

Example 3:

This example shows how one can implement a path animation using scripting.

Create a script choreography for the Translate attribute, with the following JavaScript program:

```
uvw = new r3Vect(Time, 0, 0);
p0 = new r3Vect();
p1 = new r3Vect();

// evaluate two points from the input object: one with time=0 and
// another one corresponding the current time
Input.EVALUATE(new r3Vect(0, 0, 0), R3SPACE_ABSOLUTE, p0);
Input.EVALUATE(uvw, R3SPACE_ABSOLUTE, p1);

// compute translation: delta = p1 - p0
delta = p1.sub(p0);

// we need to map 'delta' from abs space to target object's parent space
// because target objects translation is defined in its parent's space
parent = Target.GetParent();
parent.POINTTOOBJSPACE(delta);

// add delta to the target attribute
P.fadd(delta);
```

Now, bind the script choreography to desired path object, say a nurbs curve and play the animation to see how the target object follows the curve.

Example 4 - a point deformer:

A typical deformer works as follows: it first maps a point to a simple geometry, such as a rectangle. Then it use the result to evaluate a point from another, typically more complex, geometry.

For example, let's imagine a deformer that turns any planar mesh to a sphere.

Such a deformer can be represented as a level object consisting of two sub objects: a rectangle and a sphere.

```
deformator
 /      \
rect    sphere
```

Then we can use this deformer object as an input for a script choreography, with the following script:

```
// this chor must be bound to an input object
// that has two sub objects defining desired deformation

if(Input) {

    // map target point to abs (world) space
    p = new r3Vect(P.x, P.y, P.z);
    Target.POINTTOABSPACE(p);

    // fetch sub objects from the input object
    from = Input.GETSUBBYORDNUM(0);
    to = Input.GETSUBBYORDNUM(1);
```

```

// map point to from's param space
uvw = new r3Vect();
from.MAPTOPARAMSPACE(p, uvw);

// evaluate out through to's param space
to.EVALUATE(R3SPACE_ABSOLUTE, uvw, p);

// map point to target space
Target.POINTT00BJSPACE(p);

// add deformation effect to the target point
P.fadd(p);
}

```

Now save the choreography and you can later load and attach it to any target and input object. You can replace the 'sphere' with any geometric objects. For example, if you prefer a deformation that maps a rectangle to a cone, just use analytic cone instead.



Note

Never modify attributes through 'Target' object. For example, never call:

```
Target.SetMatrixTranslate(delta);
```

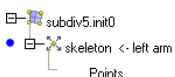
If you need to access an attribute of the target object, make the attribute animateable and access it through the 'P' variable.



Generic

Generic constructor doesn't define any animation effects by itself. Instead, it asks the associated input object (typically a geometric object, such as a skeleton) to apply the effect.

Generic constructor cannot be created directly. It is automatically created when the actual deformer object is created and bound to its target objects.

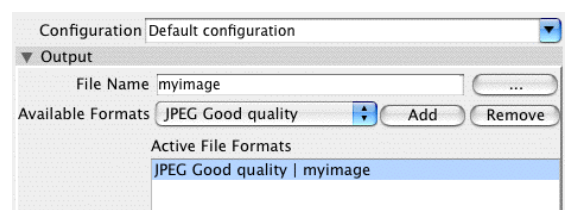


Skeleton, bend, radial deformer, 3d lattice and many other geometric deformaters are linked to the choreography system this way.

*The points of
'arm' (SDS object)
deformed by 'left
arm' (skeleton)*

File Rendering Configurations

File rendering configurations combine most settings and other data components required when rendering a 'final' image into a disk file. For example, image resolution and rendering quality settings are part of the configuration data. Often, in addition to selecting a suitable file rendering configuration, the user only has to make sure that a suitable camera object is included in the scene.



The 'Configuration' field contains a list of file rendering configurations. Selecting 'New' from the end of the list adds a new item to the list. It can be renamed and edited as appropriate, and saved to disk for later use.

The file rendering system supports simultaneous output to several files or other output targets (a preview window etc. may be also included). The following output controls are available:

- *File Name*: The name and path for the output item selected in the 'Active File Formats' list below. Changing the string and hitting the 'Enter' key updates the respective file output. The name string is also used when adding new formats to the active list.
- *Available Formats*: A list of currently loaded/created output objects. You can pick a format and click the 'Add' key to insert a new output file to the active list below this control.



Note

If a suitable format is missing, you can create new output objects at the select window's 'Output Objects' tab.

- *Active File Formats* This list shows the total output file configuration. Each item in the list is a pair of a format name and an suitably abbreviated file name. Items can be removed by activating them and using the 'Remove' popup menu. The file name part of an item can be changed by activating the item, then changing the 'File Name' string and hitting the Enter key. Note that changing the 'Available Formats' field selection does not change any item in the active list.

File formats can be added also by drag and dropping them from the select window's 'Output Objects' tab into the active format list.



Note

You cannot add the same format twice to the active list, because the items on the active list are identified by the output object name. If you, for some unusual reason, truly want to output two identical files, create a duplicate of the format object at the select window and rename it.

Settings

Rendering quality settings and post processing settings can be select as follows:

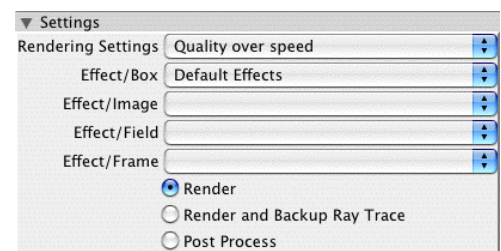
- *Rendering Settings*: The render settings object which defines ray tracing quality and other photorealistic rendering options.
- *Effect/Box*: The post processing configuration (post image) which is executed for all rendered images and image sub regions.
- *Effect/Image*: The post processing configuration which is executed for complete images. If an image is rendered by dividing it to sub boxes (e.g. when using network rendering), the boxes are processed with this configuration after assembling them into a complete image.
- *Effect/Field*: The post processing configuration for rendered fields.



Note

If field rendering is turned off, these effects are executed for all rendered images after combining possible motion samples.

- *Effect/Frame*: The post processing configuration for complete frames. Executed after possible sub fields have been assembled into a frame.

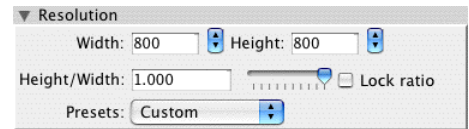


- *Render/Backup/Post Process*: With this control, you can activate rendering with an additional ray trace backup file. After full rendering, post processing effects can be readjusted and rendered multiple times using the backup file.

Resolution

These controls define the pixel resolution of the outputted images.

- *Width*: Image accuracy in horizontal direction (pixels)
- *Height*: Image accuracy in vertical direction
- *Height/Width*: Image resolution ratio.



Note

This is not necessarily the same as the image aspect ratio, which also depends on the pixel aspect ratio. Pixels are often not squares. For example, all common video standards use non-square pixels.

- *Lock ratio*: Locks the ration so that if you change one dimension, the other becomes scaled respectively.
- *Presets*: A list containing some common industry standard resolutions.

Box Rendering

The 'Render Box' controls limit the rendering to a sub area of the total image. For example, it may happen that after spending one whole day for high resolution rendering of a complex scene, the user notices that the color of a small object at the top left corner was wrong.



Instead of re-rendering the whole scene, the user can render one quarter of the total image width and one quarter of the total height around the object to a new file. The position of the new sub image is known accurately and therefore it can be pasted over the original image using a paint or image processing program.

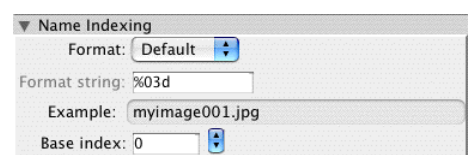
The controls are:

- *X*: The horizontal position of the top left corner of the render box (measured in pixels)
- *Y*: The vertical position of the top left corner of the render box
- *Width*: The width of the box (pixels)
- *Height*: The height of the box
- *Active*: Activates the sub box rendering feature

Name Indexing

The 'Name Indexing' controls define how the file name is generated from the actual file name and the current frame number. Rendered images are named as follows:

[filename][index].[extension]



For example:

myimage000.bmp

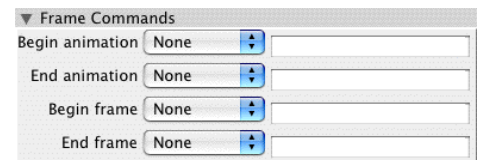
The first portion is the file name defined by the user (such as d:\mypics\test). Index is the current frame plus the base index defined through the '*Base index*' field. The number of digits in the index portion is automatically extracted from the number of frames in the animation. For example, if the length of the animation is 120 frames, the index portion contains three digits. The extension is determined by the selected file format.

The indexing controls are:

- *Format*: Selects between the default (zero padded) indexing and a customized, user defined indexing. The default indexing format is flexible enough for most cases. If a more exotic indexing convention is desired, it can be achieved by selecting 'Custom' to the 'Format' field and entering a suitable string to the 'Format String' field. The format string follows C language 'printf' conventions.
- *Format String*: This control shows the format string that is used for generating the index portion of the filename. If 'Format' is set to 'Custom', you can define an arbitrary format string to this gadget. For example, the string '%d' outputs indices without zero padding: frame0.tga, frame1.tga, ... , frame99.tga.... '%05d' would output: frame00001.tga. '%03d' outputs: frame001.tga, etc.
- *Example*: shows a complete sample filename generated by the current settings.
- *Base Index*: Animation frame indices start from zero. If you want to start output file indexing from one (frame001.tga, frame002.tga, ...), set 'Base index' to one.

Frame Commands

The 'Frame Commands' area contains four command gadgets. The commands can be defined using any of the installed scripting languages. In addition to this, DOS (or shell) scripts can also be used.



The following commands can be defined:

- *Begin animation*: This command is executed once in the beginning of animation rendering. For example, a special hardware for animation recording can be turned on.
- *End animation*: Executed once when the animation is ready. This can be used for clean up actions.
- *Begin Frame*: Executed before rendering of each frame.
- *End Frame*: Executed after each rendered frame. For example, the rendered image can be recorded to a single frame recorder.

The frame command can access the file name via the following predefined macros:

[image] - Full name of the rendered image, for example 'd:\images\test000.bmp'. *[path]* - Path portion of the file name without the last path separator, such as 'd:\images'. *[base]* - Base name (i.e. path and extension removed). For example 'test'. *[ext]* - Extension without the dot. For example 'bmp'.

For example, if the file name is d:\realsoft3d\images\test.bmp:

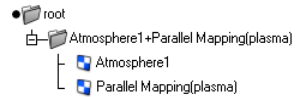
[path] - results in 'd:\realsoft3d\images' *[path][base].[ext]* is the same as *[image]* - 'd:\realsoft3d\images\test.bmp'. *[path][base].tga* results in 'd:\realsoft3d\images\test.tga'

Geometric Objects

Geometric Primitives

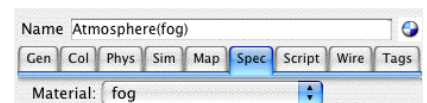
Atmosphere

The atmosphere object has no geometry. It generates a world filling volume, which can be used for global for effects. The atmosphere object allows direct material selection from the **Material** field of its property tab. The material selector shows all materials of the current library that define volumetric properties. Note, however, that since the atmosphere has no geometry, only gas materials with constant properties should be assigned from the 'Material' gadget.



You can map a gaseous material to an atmosphere object ...

If the material includes turbulence or other density variations, it is better to map the material to the atmosphere using for example parallel mapping.



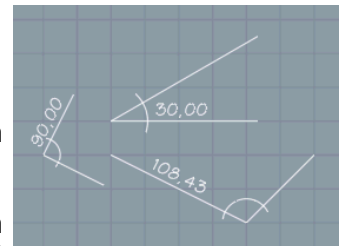
... or assign it directly as an object property

Angle Measure

The angle measure object is displays the angle between its two control lines. You can add angle measure objects to you scenes to show information about important angles.

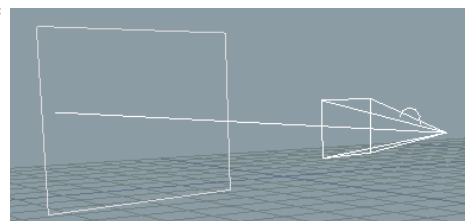
The properties of the angle measure object are:

- **Length** - The current angle
- **Text Size** - The size of numerical digits (in pixels), which display the angle in view windows
- **Threshold** - Threshold for measure drawing. Angle measures, which are seen in a very low angle (viewing direction almost parallel to the angle plane) should not be drawn, because the projected angle looks like a straight line. This threshold controls when clipping occurs. Value 1 draws all angles.



Backdrop

A backdrop object is a plane surface that automatically rotates itself perpendicular to the camera in ray trace rendering and scales itself to fill the image. It defines UV coordinates which run from (0,0,0) at the top left corner to (1,1,0) at the bottom right corner. A backdrop is a real plane surface whose distance can be adjusted using the **Distance** control on the property window's object specific tab.



A backdrop plane, such as all other geometric objects, can be textured (and shaded in general) using VSL materials. The distance can be modified using a ray hit shader, the backdrop can be translucent etc. Backdrops have some special object attributes set by default:

- They are self-illuminating (Property window's Color tab/Illumination channel)

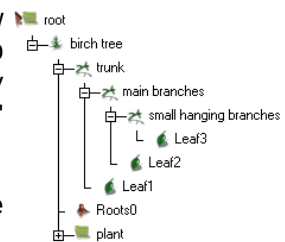
- The diffuse shading is turned off (Color tab/Color channel)
- They do not cast or receive shadows (General tab)

Branch Object and Root Object

Branch object defines how a plant grows using a number of growth parameters. It generates a set of sub branches once per 'year'. Age is a relative concept here; you can consider the units as days, if the plant grows rapidly. Sub branches follow the same growth rules as their parent, unless child branch objects are present. In other words, plant's sub hierarchy defines the structure of the generated geometry in a natural way.

A branch object can have multiple child branches. When a branch decides to grow a sub branch, it seeks for active sub branches, and only one (the first active) sub branch is invoked to grow. For example, a certain sub branch type 'leaf branch' may be active during the first 10 age units, after which another sub branch 'flower branch' becomes active.

The plant root object is identical with the branch object, except that it grows to the opposite direction. As the name suggests, it adds roots to the plant.



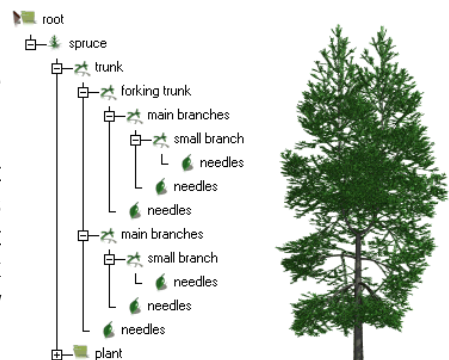
Complex nested sub branching structure

Model Type: Defines the type of the generated geometry. The options are:

- **Analytics** - Branches are built using cones, cylinders and spheres. This kind of geometry is very fast to render, and also quite memory efficient if branches are relatively straight (i.e. when the Density parameter is 0.5 .. 1.0). The UV Coordinates of this model type are limited: U component is constant zero, and V grows stepwise per geometry item.
- **Particles** - Branches are built from 2D particle segments. This kind of geometry is also fast to render and more memory efficient as analytic primitives. The real time geometry is drawn using thin unshaded lines, which makes the tree very fast to manipulate in modeling. Particle geometry does not support UV Coordinates at all in the current version.
- **NURBS** - Branches are ray traced NURBS curves. The NURBS option has no UV Coordinate support (except the native UVs of all NURBS curves), but the lowest memory consumption in ray tracing, reasonably fast rendering and very fast (wireframe only) real time drawing. NURBS curves do not currently support pointwise attributes, which means that branch diameter control is limited to the start and end points.
- **SDS** - The most detailed way to model the branches. Features: highest memory consumption, accurate UV support, possibility to apply displacement mapping etc.

Own Hierarchy: If set, the branch geometry generated by the selected object is placed into a folder dedicated for this branch type. You can then place texture maps and other material attributes, which are specific to this branch type, into the same folder.

Active by Cut: If set, the branch starts growing only when the parent branch is abnormally truncated by its Tail Cut Age or Rnd Cuts attributes. Many trees start growing a new trunk from one or two biggest branches if the actual trunk becomes cut in a storm. The new trunk usually has some special properties, such as a strong desire to grow upwards.



A spruce with a forking trunk

Activation: The age when this branch type becomes active. Inactive branches do not grow. The first active sibling branch defines the growth parameters for a new branch.

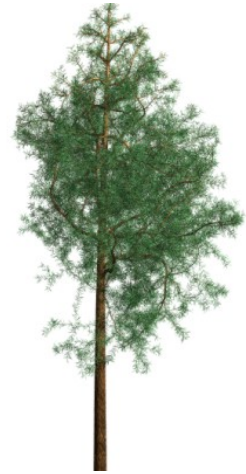
Deactivation: The end of active growth period.

Rnd Activation: Probability that the activation of the branch fails.

Age: The maximum age of this branch type. A branch's age cannot exceed the remaining growth age of its parent after the branching event. A negative value -1 automatically sets an age, which is exactly the remaining growth age of the parent after branching. Real world trees grow this way. However, it is often necessary to limit the age of sub branches of old trees, because the complexity grows exponentially by age.

Rnd Age: Randomizes branch age. The value 1.0 will set a random age between 0 ... Age.

Branching Age: The length of the branchless section in the beginning of the branch. If the value is zero, the branch starts growing sub branches immediately at its root. Some trees, such as pine, grow a long trunk, which has sub branches only at the topmost part, because oldest branches die and fall down. Use this parameter to simulate such trees.



An old pine tree has only the top branches left. Branchless value of the trunk is 8 years.

Tail Cut Age: Cuts the branch tail after the given number of years.

Rnd Cuts: The probability that the branch ends at a branching point. Only the sub branches continue their growth. This kind of growth happens when a branch gets damaged for some reason (gardener cuts the branch, a disease or insect kills it etc.)



The tree on the right has a high Random Cuts value. Its branches often split into two or three side branches.

Sub Count: The number of sub branches, which start growing from a branching point.

Rnd Count: Randomizes the count. Value 1.0 selects a random number from the interval 0 .. Sub Count.

Sub Thickness: The relative thickness of a branch vs. its parent.

Rnd Thickness: Randomizes the relative thickness value.

Thickness/Age: Changes the relative thickness by the age when branching happens. If the value is high, the relative thickness of oldest branches is smallest. For example, a spruce has a thick trunk. Even the strongest main branches are thin compared to the trunk. However, the trunk gets thinner towards the top, and the upmost branches are quite as thick as the trunk from which they grow.

Bumpiness: Moves the points of SDS modeled plants randomly, making the surface appear bumpy. Other geometry types are more limited with respect to this attribute: Analytic and Particle modeled trees allow only rapid variation of the diameter. Nurbs trees do not currently support Bumpiness.



Bumpy SDS branch

Density: Defines how densely the growth parameters are evaluated and new geometry added to the tree. The smaller the value, the stronger bends branches can make because of gravity and other deformation factors. The drawback is that densely evaluated geometry consumes more memory.

Length: Defines the growth speed of the branch. The value is relative to the of the parent branch: if Length equals 1, the sub branch grows the same length per year as its parent branch.

Rnd Length: Random variation in the length. A value 0.5 changes the length -50% +50 % randomly.

Length By Age: Defines how much shorter/longer the branch segments get every year. The value is multiplies the length, meaning that the factor 1.0 keeps the length the same. Many plants first grow rapidly higher to reach the light from the shadow of bigger plants. When they reach a sufficient amount of light, they target their growth energy to spread branches sideways.

Angle: The angle between the branch and its parent.

Rnd Angle: Random variation of the branch/parent angle.

Angle/Age: Angle reduction by age. Usually oldest branches have a greater angle with the trunk as the younger branches at the top, because the weight of the big branches bends them down.

Twist: Rotates sub branch directions around the branch. Twist option creates spirial wise branching patterns.

Rnd Twist: Randomizes the twist parameter.



The lower branch twists sub branches around itself

Gravity: Guides the branch orientation by gravity. The direction of gravity is defined by the plant's root object's z axis (blue axis line). The higher the gravity, the more branches will hang down. A negative gravity value makes branches to grow upwards (to simulate light attraction etc.).

Gravity/Age: Defines how the effect of gravity changes by the age. The actual gravity is the sum of Gravity and Gravity/ Age parameters. For example, a branch end may seek light and grow upwards, whereas a branch start can bend down because of the total mass of the branch.

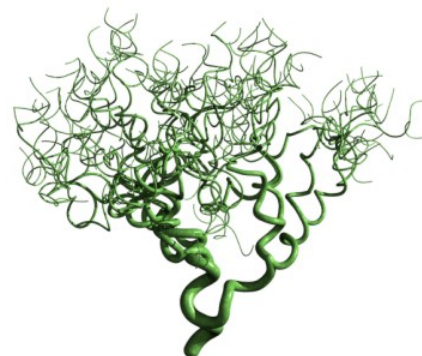


Positive and negative gravity

Side Shift: Turns the branch end to the side. By combining Twist and Side Shift Parameters, you can create spiral-like growth behaviors.

Deformation: Bends the branch randomly. All natural plants have some deformation. High value such as 3 bends the branch strongly.

Deform Scale: Defines how rapidly deformation bends occur. The higher the value, the quicker deformation turns growth direction. A low deform scale value will bend neighbor branches in a similar way.



Side shift = 0.5, Twist=1.0, some deformation

Horizontal Spread: Guides the branch to spread out horizontally around its parent branch. Horizon direction taken from the blue z axis of the object space of the plant root level. Many plants grow this way: the purpose is to catch as much light as possible. Note that will full horizontal spreading level 1.0 and Sub Count greater than 2, branches may overlap.



The branches of the tree on the left grow to random directions. Horizontal spreading makes the branches of the tree on the right to grow sideways.

Spread by Age: Changes the horizontal spread by the age when the branch started to grow. The actual spread parameter is the sum of the constant spread and the age dependent spread. If Horiz. Spread=0, Spread by Age=1.0, full horizontal spreading is achieved at the end of each branch. If Horizontal Spread is 1 and Spread by Age -1.0, spreading reduces towards the ends of branches.

Branch Profile Attributes

These attributes define how the diameter of the branch changes towards its end point. You can define the profile using a constant tapering factor with some randomization, or using a profile curve, or as a combination of both controls, in which case the tapering factor is the product of the two controls.

Note: NURBS curves do not support pointwise attributes in the current program version. Branch diameter control is limited to the start and end points with this geometry type.

Tapering: Defines how much each branch section gets thinner during one age unit (from a branching point to the next branching point). Tapering factor 0.2 means that the end radius is 20 % less than radius at the start.

Rnd Taper: Randomized tapering. A value 0.1 changes the radius randomly -10 % ... +10 % at every age step.

Profile Curve: The diameter profile of the branch (excluding the effect of the tapering factor). With the curve, you can define accurately the shape of a tree trunk, or the branches of a cactus tree, which branches actually thicken after the branching point.



A cactus has an unusual branch thickness curve. Branches are thickest in the middle.

Leaf Density Attributes

U: The leaf density around the branch.

U: The leaf density per age unit along the branch.

Density Curve: The density of leaves by age. This curve controls all leaf types in a deterministic manner; the density is the product of U and V factors and the curve value. Leaf objects have another curve, which controls the density of each individual leaf type by defining a stochastic distribution for it.



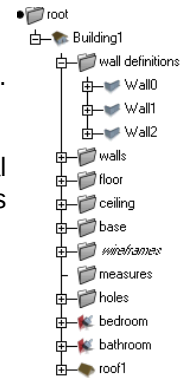
Flowers are more rare than ordinary leaves

Building

The building object constructs a complete building from the parameter lines. It manages roof, wall and room sub objects to perform this job.

The hierarchy of the building object has the following sub levels:

- wall definitions: Contains the wall parameters. The first sub wall object is the outer wall. Other walls split the building into rooms.
- walls: the wall geometry which the building object generates. You can place material attributes into this folder. Editing the wall geometry items while the building construction is enabled makes no sense, because next construction refresh will remove all changes.
- floor: if the floor feature is enabled, the floor surface is placed here.
- ceiling: if the ceiling feature is enabled, the ceiling is placed here.
- base: if the base feature is enabled, the base plate is stored here.
- wireframes: contains temporary geometry which is added to achieve a clean wireframe representation.
- measures: the measure lines showing the dimensions of the building are stored here, when the respective option is turned on.
- holes: the hole tool places the hole parameter data (tagged rectangles) to this folder. Note that a hole can be anywhere in the hierarchy of the building (embedded into a window or a door object) and it still works.
- room1, room2: room objects, which decorate room surfaces locally.
- roof: the roof creator object. If present, the building tool automatically extracts the roof edge curve and updates the parameters for the roof respectively.



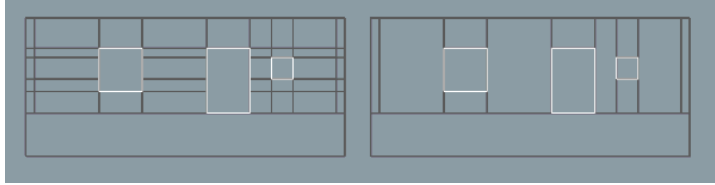
The structure of a building object

The building object has the following options available in its property gadget (Property window/Spec tab):

Wall Geometry options:

- Build: enables automatic construction. Generated wall geometry can be edited manually only after Build is turned off. Turning Build on removes manual changes. When the option is enabled, only the parameter data is saved to disk files, which usually makes the file size small.
- Floor: enables automatic floor surface construction.
- Ceiling: enables automatic ceiling surface construction.
- Auto Texture: if enabled, the building object automatically adds and updates material mapping objects for walls, ceiling, floor and base.
- Geometry: SDS geometry has a better OpenGL preview, but is somewhat slower to render. It is easier to make Boolean operations with Analytic geometry.
- Height: height of the walls. Note that sub walls have an option 'Extend to Roof', which can override the common base height.

- **Uniform Division:** each wall section is divided vertically using the same division. This generates a denser SDS mesh, which is usually easier to edit manually.



A wall with 3 holes. Uniform wall geometry division on the left, minimal division on the right.

Base options:

- **Shape:** either a simple rectangular plate or a polyhedron, which matches the outer wall shape.
- **Height:** the height of the base. The base extends down from the walls. Changing the height does not lift the roof higher.
- **Offset:** shrinks the base plate inwards from the outer walls.

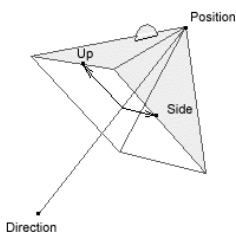
Measure Lines:

- **Measure Lines:** None, Left, Right or Both. Selects where the measure lines are shown.
- **Offset:** the distance from the wall polylines to the measure lines.
- **Text Size:** defines the font size (in pixels) for the displayed dimension values.
- **Arrow Size:** size of the end arrows (in pixels)

Camera Object

A scene can include any number of camera objects, in which case the 'Current' attribute indicates the current active camera. The Current attribute can be animated so that the current camera may change during the animation.

The geometry of camera object is defined by four points.



The Center specifies the position for the camera. The Direction point defines the aim point for the camera. The focal axis can be derived from the two by subtracting the Position from the Direction point. Also the focusing distance, which is needed for the depth-of-field effect, is defined as the distance between the two points. The two other points, 'Up' and 'Side', define the opening angle for the view pyramid in both directions. The image aspect ratio, for example, can be derived from these geometry points.

The geometry can be skewed, in which case the film surface is no longer perpendicular to the focal axis.

Realsoft 3D supports both perspective cameras and orthographic cameras. This property is controlled via the 'Projection' attribute. Orthographic cameras are often used for modeling purposes, or to simulate distant cameras.

In addition to geometric attributes, a camera object defines many other attributes well known from their real counterparts.

Film Sensitivity acts like a general brightness control.

Depth-of-field effect depends on the properties of the camera. The following attributes can be used for controlling it:

- **Depth Of Field** - on/off
- **Focal Length** - the focal length of the lens
- **Fstop** - fstop value

- **Focusing Distance** - the distance between the aim point and the position point.

In real world, the lens flare effect depends on the properties of the camera. This is also the case with Realsoft 3D. The lens flare effect is automatically generated according to the properties of the light sources and the camera. Camera object defines an attribute 'Lens Flare', which specifies the name of the lens flare post effect for the camera.

The camera object also supports a 'Flash light'. The color of the flashlight can be defined freely. Black color turns the flash light off.

Camera Object Property Gadget

Perspective/Orthographic: The projection type of the camera.

Current: Selects the 'Current' camera, which is used in camera tracking and in file rendering.

Vertical Film: Activates a special film projection which keeps vertical lines parallel.

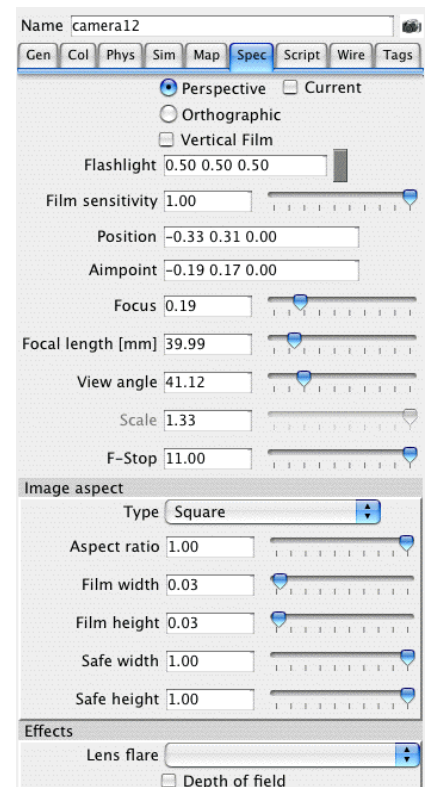
Flashlight: Defines the color and intensity of the camera flashlight.

Film sensitivity: Controls the overall brightness of images rendered using this camera. The higher the value, the more sensitive the film is, becoming more exposed so that the picture appears lighter.

Position: The location of the camera in 3D space.

Aimpoint: The point camera is directed to. Defines also the sharp distance for Depth of Field.

Focus: The distance from the camera position to the aim point. When using depth of field, the image appears sharp at this distance.



Focal length: Defines the opening angle of the camera.

View angle: The same as above, expressed as an angle value.

Scale: The same as above, expressed as inverse of width of the visible area at 1.0 meter distance. Useful in orthographic projection. The scale value describes how much the camera magnifies the image.

F-Stop: The lens opening.

Image Aspect Frame

Type: A list of predefined camera aspect ratios.

Aspect ratio: The current image aspect ratio (height/width of visible area). Note: this value is independent of pixel resolutions and pixel aspect ratio. Set them separately for example in the File Rendering window.

Film width and Height: Film size also controls the aspect ratio and affects depth of field.

Safe Width and Height: You can define a safe sub region using these controls. The safe region is part of camera wireframe and is used as a visual guide only (does not affect rendering). When tracking a camera view, you can use an appropriate safe region guide to verify that critical objects fit into the image area that is always visible. For example, borders of TV screens always hide part of the received image signal. (As a general rule, 0.9 is "safe motion", 0.8 is "Title safe".)

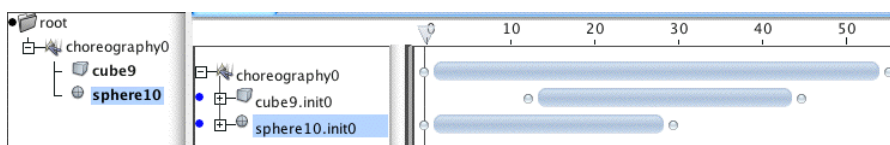
Effects Frame

Lens flare: Activates automatic camera/light source lens flare effect. You can edit the properties of the flare from the post particle effects tab of the select window: activate the same effect there and open the property window.

Depth of Field: Activates camera defined depth of field effect. The DOF properties can be edited from the post image effects tab of the select window. Select the DOF effect that is used in the current post effect configuration and change its properties. Note: this setting is relevant only when the post effect configuration includes a DOF effect with the 'Camera Defined' option enabled.

Choreography

The choreography object is a level object which provides hierarchical time line manipulation. You can place several animated objects inside a choreography level and control the timing of them all through a single time line.



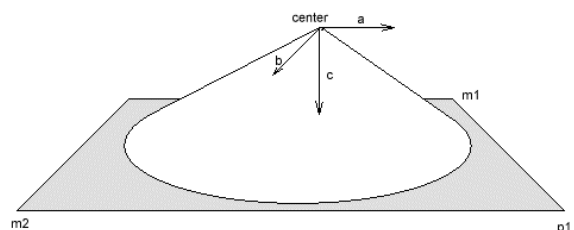
The attributes of the choreography object, available at the Spec tab of the property window - are the same as those which can be manipulated through the choreography editor:

- **Begin** - The start of choreography object's time line.
- **End** - The end of choreography object's time line.
- **Frequency** - Repetition frequency. For example, value 2 will repeat the sub animation events twice during the time line.
- **Phase** - Shifts the phase of the included animations.
- **Weight** - Choreography weight. Controls how strongly the choreography affects the object.
- **Periodic** - Periodic time evaluation for creating cyclic motions.

Cone

An analytical cone can be derived from an ellipse by a third axis 'C'. Three points 'p', 'm' and 'n' are used for defining so called 'clipping plane', which trims the infinite cone surface to a desired finite size.

For example, if the length of all the three axes is the same, a circular cone with 45 opening angle is achieved.



The cone supports efficient volume enclosure test, which makes it especially useful object in Boolean operations.

Constraint

The constraint object serves scripting purposes. Normal objects can access their own attributes at the Scripting tab. The constraint object can access attributes of all objects that were selected at the constraint creation time. For example:

- Create three spheres and create a simple animation where all the spheres move here and there.

- Select the spheres
- Create a constraint (pull down menu Tools/Construction/Create Constraint)
- In the Spec tab of the property window, set Command Language to JavaScript and enter the script below.

```
if(P.y < 0)
    P.y = 0;
```

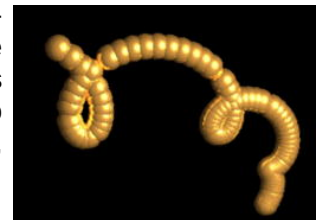
The 'P' variable refers to the current animation channel, Translate attribute in this case.

The constraint object defines the following variables:

- Target - target object. You can read any target attributes simply by calling appropriate Get method. However, you should modify target objects through the 'P' variable. This results more reusable code.
- Self - constructor object itself.
- Time - current time
- Type - the type of the target attribute (as defined in r3typids header file)
- Tag - attribute identifier, as defined in the class header file
- Index - attribute index, or -1 for regular attributes

Creator

The Creator object is a level object that generates duplicate copies of its sub-objects during an animation. The level that contains the Creator must have the Simulation property turned on. There are many uses for creator objects: bullets being shot from a gun, bees leaving a hive, water dripping from a faucet, just to offer a few examples. Creator objects can be animated using keyframes or paths, just like many other object types.



The **Active** checkbox turns creation on and off. You can animate it to achieve object creation at suitable intervals.

A creator objects moves along a path and shoots spheres around

The **Parent Level** attribute specifies how many hierarchy levels above the creator the emitted objects are inserted. The default is 0, which causes objects to be inserted at the same hierarchy level that contains the Creator object.

The **Creation Interval** attribute defines the time interval between the creation of emitted objects; that is, it defines how rapidly the Creator emits objects. The smaller the value, the more objects are created. The time interval is specified in the current time units (defined through the Options window). So, for instance, if your time units are frames, setting the creation speed to 15 will cause a sphere to be emitted every 15 frames, or two times per second for an NTSC video system. If the **Channel** gadget below the creation interval gadget includes a channel selection, materials that affect the creator are evaluated at each sample object position. The evaluated channel value is multiplied with the creation interval value. The result is used as the actual sample specific creation interval.

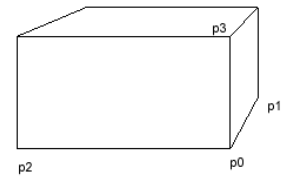
The **Life Time** attribute defines the time interval that an emitted object is active before being destroyed (i.e. its lifespan). The smaller the value, the shorter the object's life. The time interval is specified in the current time units (defined through the Options window). So, for instance, if your time units are frames, setting the Life Time to 15 means a sphere is active for 15 frames (0.5 seconds in NTSC video) before being destroyed. If the **Channel** gadget below the life time interval gadget includes a channel selection, materials that affect the creator are evaluated at each sample object position. The evaluated channel value is multiplied with the life time value. The result is used as the actual sample specific life time.

The **Velocity** attribute defines the rate at which the created objects move, in meters per second. Since this attribute is a vector value, you define velocities along the three axes. To get motion only along the X axis, you would enter a value such as "0.5 0 0". Velocity is defined in the Creator object's space. The **Channel** gadget defines a material channel whose contents is multiplied with the given velocity value to obtain the sample object specific velocity.

The **Spin** attribute defines the rate at which an emitted object spins. Since this attribute is a vector value, you define the spin along the three axes. If the associated **Channel** gadget includes a channel selection, materials that affect the creator are evaluated at each sample object position. The evaluated channel value is multiplied with the spin value. The result is used as the actual sample specific spin.

Cube

An analytical cube is defined by four corner points, defining three edges (or three faces) for the cube. Other faces are computed internally and they are always coplanar to their opposite counterparts.

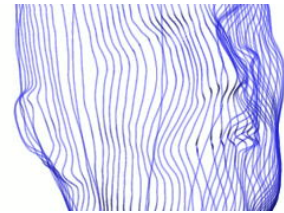


The rounding attribute of cube defines an efficient render time edge rounding. For the sake of speed and memory use, the rounding is shown only in ray tracing.

Curve Manager

The curve manager object creates a set of NURBS curves by evaluating its parameter objects.

- **Displacement** - Moves the curves away from the parameter surfaces by displacing the points with the defined distance to surface normal direction.



Note

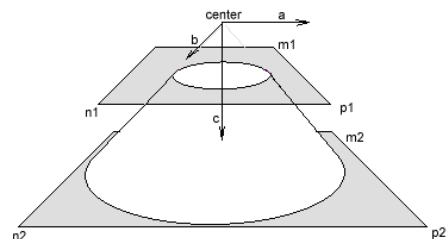
Also negative values are allowed.

- **Channel** - If a channel is selected here, materials affecting the parameter objects are evaluated. The value of the selected channel is used to displace curve points.
- **U Curves** - If set, curves in the U parameter direction are evaluated.
- **V Curves** - If set, curves in the V parameter direction are evaluated.
- **Curves** - The density at which curves are evaluated.
- **Points** - The density of points in created curves.

Cutcone

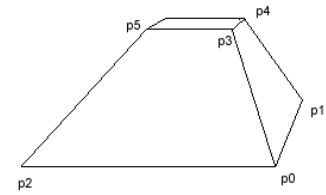
A cut cone object can be derived from a cone by introducing a second clipping plane.

Also the cut cone supports a quick inside / outside test, which makes it efficient in Boolean operations.



Cut Pyramid

A cut pyramid is defined by two rectangles: the top and the base for the pyramid. All faces of the pyramid must be planar 2d surfaces and the top and bottom faces must be coplanar.

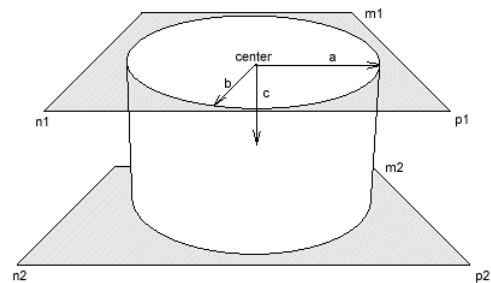


Cylinder

An analytical cylinder can be derived from an ellipse by introducing a third axis attribute 'C', the axis of the cylinder.

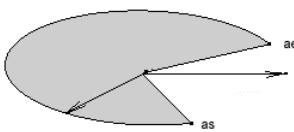
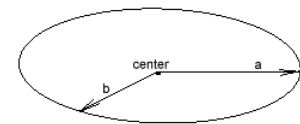
To get the infinite cylinder surface trimmed to a finite size, the cylinder object also defines two so called 'clipping planes'. These clipping planes define Boolean operations which clip out all points which don't reside between them. A clipping plane is defined by three attributes: one corner point (p) and two vectors specifying the associated edges for the plane (named as 'n' and 'm').

A cylinder performs an efficient volume test, which makes it useful object in Boolean operations.



Ellipse

An analytical ellipse is defined by a center point and two axes. The axes don't have to be perpendicular to each other. If the lengths of both axes are equal and the axes are perpendicular to each other, a special case called a 'circle' is achieved, in which case the lengths of the axes are the radius of the circle.



Ellipse, as well as nearly all other geometric shapes derived from it define an attribute called 'Sector'. When set, two additional attributes 'Start Angle' and 'End Angle' can be defined to achieve sector like shapes. Internally, all angle attributes are represented as radians, but the user interface allows use of other standard units as well.

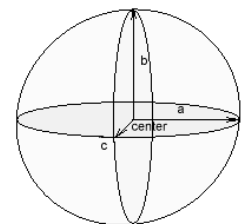
Ellipsoid

An analytical ellipsoid is derived from an ellipse by introducing a third axis 'C'.

When all the three axes have equal lengths and they are perpendicular to each other, the shape is a sphere.

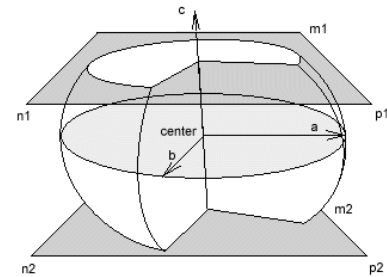
The sphere supports an efficient inside / outside test, which makes it especially useful object in Boolean operations. It is also very quick to render in ray tracing.

An ellipsoid doesn't support Sector attribute. If this option is needed, use an Ellipsoid Segment object instead.



Ellipsoid Segment

An ellipsoid segment differs from the ellipsoid object in that it supports two clipping planes as well as the sector attribute.



Extrude

The extrude manager object builds an extruded NURBS mesh system from the parameter NURBS curves. The following options are available:

- **Radius** - The bevelling radius. Measured in millimeters.
- **Depth** - Depth of extrusion in current measuring unit.
- **Bevelling** - The bevelling type. The following alternatives are available:

- *None* - Extrusion creates sharp 90 degree edges.

- *Sharp Edge* - Edges are bevelled with a straight line cut at 45 degree angle.

- *Rounding* - Edges are rounded smoothly along a circle arc.

- *Groove* - Edges have groove like round bevelling.



- **Bevel Out** - The beveling is created by expanding the parameter curves. This creates fatter objects.

- **Independent** - If set, extruded surfaces from each parameter curve are collected into a separate hierarchy level. If cleared, all surfaces are placed to a same level.



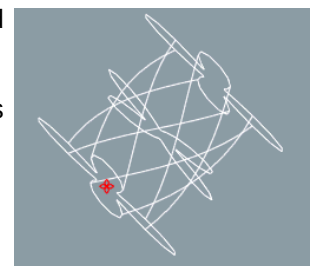
*Inwards bevelling on the left,
Bevel Out applied on the right*

Fan

The fan object is used in fluid simulations. It generates a fluid environment and produces a flow with the following controls:

Turbulence: Amount of turbulence in the flow. If turbulence is zero, the flow has the same constant direction everywhere.

Radius: The radius of the fan object. Radial falloff below uses the radius.



Radial falloff: This setting defines how the flow reduces away from the fan axis.

- *Constant* - The flow does not depend on the distance from the axis.
- *Linear* - The flow decreases to zero at the outer fan edge at a constant speed.
- *Quadric* - The flow speed is defined by a 2nd order parabola: $\text{speed} = k \cdot d^2$, where d is distance from the outer edge towards the axis. This kind of flow appears in real world water pipes.

Axis falloff: Defines how rapidly the flow decreases in the direction of the fan axis. The falloff is symmetric in front of and behind the fan.

- *Constant* - The flow does not depend on the distance measured along the fan axis.
- *Linear* - The flow is inversely proportional to the distance: $\text{flow} = \text{maxflow} / (c + k \cdot \text{distance})$



Note

The actual flow speed and other fluid properties can be defined from the Physics tab of the property window.

Field Evaluator

The field evaluator object is capable of evaluating useful material and physical properties at its location in 3D space. Evaluation follows general hierarchy rules of material mapping i.e. only materials mapped to the evaluator object or its parent levels are considered. Evaluated properties can be used as parameter information for intelligent animations. Usually the field evaluator object itself is not the target of animation or material mapping, but it is placed into the hierarchy of other objects to examine their properties.

The **Value** channel shows the result of evaluation of the material channel defined in the **Channel** selector below it. All surface property channels and user defined channels are available for evaluation.

Velocity, **Spin**, **Acceleration** and **Angular Acceleration** show the respective physical properties of the Field Evaluator object during animation playback. In addition, you can use them to enter key values when key framing objects which have been parametrized with the field evaluator attributes.

File Instance

The File Instance object instances specified Realsoft 3D project file (.r3d). A file instance object can be transformed or animated similar to any regular geometric object. However, it cannot be single point edited or morphed.

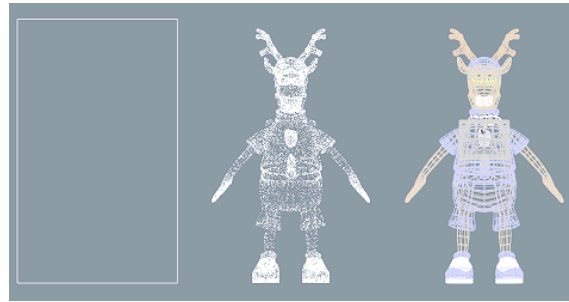
File instancing often saves memory and speeds up real time rendering when building complex scenes. Another important application is related to project management. Several users can edit and improve components of the same scene. The results are automatically combined via file instancing.

The attributes of the file instance object are as follows:

- **File** - The name of the .r3d file to be instanced
- **Geometrics** - Load in geometric objects from the file
- **Load Geometry** - Load in geometry for accurate real time rendering
- **Materials** - Load in material objects
- **Effects** - Load in particle post effects

- **Render as instance** - Activates instancing in photorealistic rendering.

If the Load Geometry option is set, the full geometry is loaded and kept in memory for real time rendering purposes. In this case, a file instance will not offer any memory saving compared to using a regular object. If Load Geometry is disabled, only a simpler visualization is available. This often saves memory and speeds up real time rendering.



Three ways to visualize file instances: bounding box, point cloud and full wireframe

The alternatives for simpler visualization are:

- If Wire Density is 1, the object is drawn as a simple wireframe bounding box.
- If Wire Density is greater than 1, the object is drawn as a point cloud. The Wire tab's Curve Quality and Wire Density controls together define the density of the point cloud.



Note

File Instancing resolves material and particle effect references primarily inside the instanced file. For example, if the user has created a wooden chair and adds a file instance of it into a new project, the wood material is first searched from the instanced file. If it is not found, then the wood is searched from the actual material library of the project.

If the Render as instance option is set, the render engine does not convert all file instances of the same target file to real objects but performs the instancing transformation during ray tracing. Only the first file instance is stored in memory; other items will transform the same geometry data. This typically slows down rendering by 20%-30%, but saves memory. For example, it is possible to model and render a forest by repeating thousands of instances of a dozen of basic tree shapes.

Applying render time instancing to simple basic shapes does not save memory, but actually can increase the memory use, because the transformation matrices require also some memory space.



Note

Render time instancing has some limitations. Currently render time instancing does not allow nested or scaled instances. Only translation and rotation are allowed. The program automatically disables render time instancing if these conditions are not set. Secondly, the applied geometric transformation may confuse complex VSL shaders which refer to 3D coordinate information.

Font

The font object creates a set of NURBS curves from a defined text. The font curves can be extruded to 3D object using other NURBS tools, such as the Extrude tool.

The font object has the following options:

- **Text** - A text string from which the curves are generated.
- **Name** - Selects the font typeface.

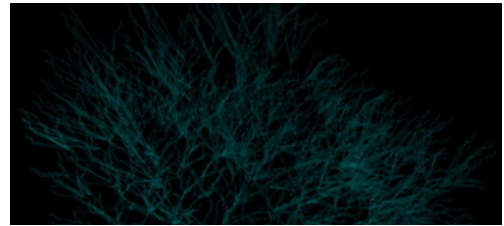


Fractal Tree

The fractal tree object creates a branching structure by duplicating a parameter curve.

The following options are available:

- **Random** - A random seed for fractal computations.
- **Age** - Controls the amount of sub branches. Full integer steps create new sub branches and the decimal part evaluates the respective sub section of the latest branch along the parameter curve.
- **Branches** - Number of sub branches per branching point.
- **Branch Angle** - Angle between branches. The number of branches sets a maximum limit for the angle. For example, if the sub branch count is 3, the angle cannot be greater than 60 degrees (a mathematical fact).



Gravity

The gravity object generates a constant gravity field that does not depend on the distance. It is useful when simulating real world gravity effects i.e. when the scale of events is so small that the gravity can be considered constant.

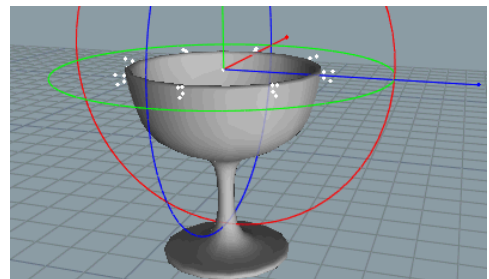
A gravity object includes two points, which define the direction of the gravity field. The length of the gravity line is irrelevant: the **Strength** of the gravity is controlled using the value at the property window 'Spec' tab. The default value 9.82 m/s^2 is approximately the gravity on the earth surface.

Group

A group object stores a set of links to geometry points. You can store a complex point selection to a group. Applying modifications later to the defined set of points becomes easier: you can modify the group instead of modifying the points directly.

You can also use the group for selecting and deselecting the points afterwards; the control bar shows the required tools as soon as you select a group object.

A group can refer to points of a single object or to multiple objects.

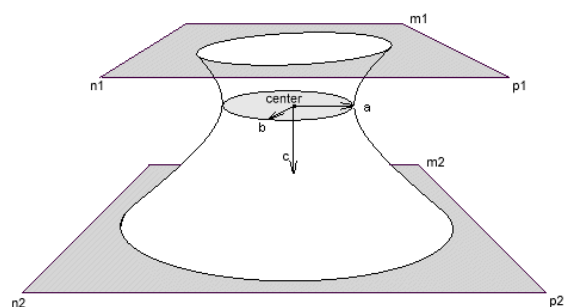


A group object for controlling the upper edge of a wine glass

Hyperboloid

A hyperboloid is defined by a center point, three axes and two clipping planes.

The object supports Sector attribute and an efficient inside / outside test.



Input Plane

The input plane object is a plane in 3D space. It stores the orientation of a modeling plane (=view's input plane) and the grid origin. You can place input plane objects to those areas of the scene where modeling actions are happening frequently. You can then set the modeling orientation and grid origin quickly by drag and dropping a suitable input plane object from the select window into a view window. You may also apply the auto focus tool to zoom into the region defined by the input plane square.

For example, when modeling a room interior, it may be a good idea to put an input plane object on the ceiling, on the walls and on the floor.

The input plane property sheet contains controls for plane orientation (HPB angles) and the position of the grid origin included in the plane. The **Current** property is used whenever a view window is set to Lock to Input Plane state. When the current input plane is modified, the view updates automatically.

Instance

The Instance object duplicates geometry of another object. Instance stores only the changes made to the original object. Therefore, it is a memory efficient way to create multiple copies of a same geometry.

Instance property gadget has the following options:

Ignore matrix - When set, the original object can be transformed (move, rotate, scale) after instance creation without changing the instances. If the option is cleared, instances take into account all object transformations of the original object and its parent levels. Which way is better, depends on the case. Even when the option is set, it is still possible to modify all instances simply by performing the desired transformation to sub objects of the original object or to its actual points.

Render as instance - activates instancing in photorealistic rendering. The render engine does not convert instances to real objects but performs the instancing transformation during ray tracing. This typically slows down rendering by 20%-30%, but saves memory. For example, it is possible to model and render a forest by repeating thousands of instances of a dozen of basic tree shapes.

Applying render time instancing to simple basic shapes does not save memory, but actually can increase the memory use, because the transformation matrices require also some memory space.



Note

Render time instancing has some limitations. Currently render time instancing does not allow nested or scaled instances. Only translation and rotation are allowed. The program automatically disables render time instancing if these conditions are not set. Secondly, the applied geometric transformation may confuse complex VSL shaders which refer to 3D coordinate information.

Interpolator



The Interpolator object is a level object which produces interpolated copies from its sub objects at photorealistic rendering. For example, you can model 1000 sample hairs and render them as 100 000 hairs.

It is much faster and easier to control the small sample object set as the real amount of hairs. Also, project file size does not grow too much. For many simple geometries, modeling requires much more RAM memory as the rendering. Therefore, the interpolator makes it possible to render much more complex objects.

An interpolation hierarchy example:

```
interpolator
  nurbs curve 1
  nurbs curve 2
  ...
```

Interpolation happens between direct sub objects of the interpolator level. Therefore, an interpolator must have at least two sub objects to function. Interpolated sub objects should have identical geometrical structure. In the example above, each nurbs curve must have the same number of points.



Note

The object interpolates only the geometric attributes of its children.



Vegetation modeled using the interpolator object

The interpolator controls are:

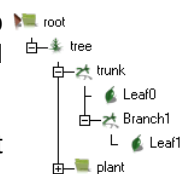
- **Count** - Number of additional render duplicates between each two consecutive sub objects.
- **Bilinear** - Selects bilinear interpolation over consecutive object quadruples. Linear interpolation places duplicates along a straight line between each object pair. Bilinear interpolation places the copies into a rectangular area defined by 4 objects. Therefore, bilinear interpolation method usually fills surfaces better than linear interpolation.
- **Closed** - Selects closed interpolation. Interpolation continues from the last sub object back to the first sub object. The option for is useful for creating circular object groups.
- **Min and Max Distance** - Interpolation happens only when the objects are within the defined range. It does not make sense to interpolate hair between dog's tail and its ear, because the hairs would float in the air.
- **Random** - Additional random variation for interpolated copies. The value defines a maximal random displacement in the current measuring unit. For example, a random value 0.1 displaces points so that each point moves to a random position that is not further than 10 cm away from the original location.

Leaf Object

Leaf object generates leaves and needles along the surface of its parent branch object. To create a leaf, select a branch object or some of them and hit the Leaf tool button in the tool control bar, or select the pull down menu Methods/Branch/Leaf.

Note that leaves are generated only for the direct parent object - parents of parents are not taken into account.

A branch object can have multiple leaf children. All leaf objects are considered active every time the parent branch object determines to insert a leaf. For example, a branch of a rose can have three sub leaf objects: a 'spike', a 'flower' and a 'leaf'.



One leaf object adds leaves for the trunk and another one for the branches

Leaf object has the following attributes:

Type: Defines the type of leaf geometry. The choices are:

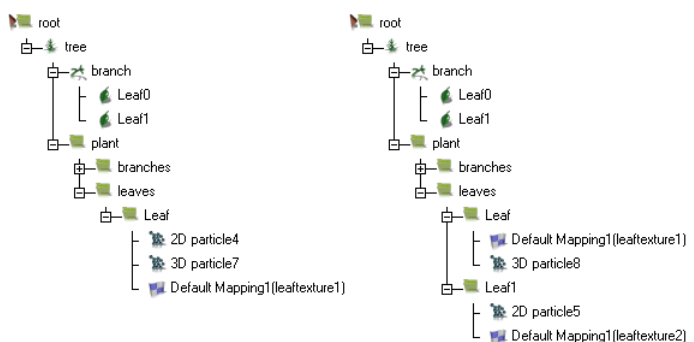
- **Copy** - leaves are generated by copying the sub objects of the leaf. If a leaf has several sub objects, the copy is selected randomly from the sub object list.
- **2D Particles** - Leaves are rendered as ray traced 2D particles. 2D particles are always symmetric by the axis, and therefore this option is mainly suitable for representing needles and spikes. The Size attribute of a 2D particle leaf defines particle's length, its radius at the start and the radius at the end.
- **3D Particles** - Ray traced 3D particle leaves. Each leaf is an elliptic sphere. Thin long particles look like needles, wider flat particles look like leaves.
- **Particle Quads** - This option renders each leaf as a planar UV mapped quadrangle. By applying a suitable color texture and a clip map, natural looking leaves can be achieved.
- **None** - Turns leaf creation off (for temporary trunk/branch editing actions etc.).



Leaf types from left: Copies of Nurbs modeled curved surfaces, 2D particles, 3D particles and clip+texture mapped flat quads

Own Hierarchy: If set, the actual leaves generated by this leaf object are placed into a separate folder. This can be necessary when the plant object includes multiple leaf objects and you want to attach a unique material map to each type of generated leaves.

Size: Size of leaves. The three values represent length, width and thickness. Thickness is irrelevant for flat quadrangles. For 2D particles, the values define length, start and end radius.



The plant on the left has all its leaves in a same folder. On the right, the Own Hierarchy option places the leaves into separate folders so that they can be textured individually.

Random Size: Randomizes the size definition components. The random factor is multiplicative: a value triplet (1, 1, 1) will give each particle a random size between zero and the defined full size. A value triplet (0.5, 0.1, 0) varies length by 50 % randomly, width by 10 %, and thickness will get no random variation at all.

SeparateRnd: If set, an independent random value scales each leaf dimension. An example: an orange tree has 3D particle leaves as fruits. The maximum leaf size (radius of oranges) can be 0.05 0.05 0.05. Because some oranges are big, some small, the Rnd Size can be (0.5 0.5 0.5) for 50 % size variation. Separate Rnd must not be set, because it would make some oranges long and thin, some flat.

Leaf Angle: The angle between the leaf and the branch from which it is growing. Note: the angle around the branch depends on several other attributes of the parent branch and the leaf itself, such as twist and horizontal spread.

Random Angle: Randomizes the angle between the leaf and its parent branch. Randomization is additive: a random value 1.0 adds a random angle variation between -90 ... +90 degrees to the regular leaf angle.

Horizontal Spread: Guides the leaves to spread out horizontally around the branch. Horizon direction taken from the blue z axis of the object space of the plant root level. Many plants grow this way: the purpose is to catch as much light as possible. Note that will full horizontal spreading level 1.0, leaves may start overlapping unless there is a sufficient amount of position randomization.



Horizontal spreading increases towards the end of the branch

Spread by Age: Changes the horizontal spread by the age when leaf started to grow. The actual spread parameter is the sum of the constant spread and the age dependent spread. If Horiz. Spread=0, Spread by Age=1.0, full horizontal spreading is achieved at the end of each branch. If Horizontal Spread is 1 and Spread by Age -1.0, spreading reduces towards the ends of branches.

Random Position: Randomizes the position of leaves. If the value is zero, leaves will grow in regular rows and columns along the branches. Full random level 1 places the leaves to totally random positions.

Gravity: Guides the leaf orientation to take gravity into account. The direction of gravity is defined by the plant's root object's z axis (blue axis line). The higher the gravity, the more leaves will hang down. A negative gravity value makes leaves to grow upwards (to simulate light attraction etc.).



Regularly and randomly positioned needles

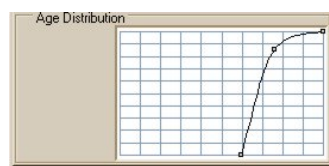
Age Distribution: The distribution of this particular leaf type in the plant by age. A rapidly ascending curve that starts from $x=0.9$, $y=0$ and ends at $x=1$, $y=1$ will guide the leaf to grow only at the end of branches. For example, plant flowers can be made to grow only at branch ends this way.



The curve defines a stochastic distribution for the event that the leaf grows at a certain age. For example, if the curve travels at the 50% probability level ($y=0.5$), about one half of the attempts to grow a leaf of this leaf type will succeed.

3D particle leaves are spread sideways using a high horizontal spread value, and then pulled down using some gravity

Note: each branch object has top-level controls Leaf Density U, Leaf Density V and Leaf Density Curve. These properties control all sub leaves the same way. The Age Distribution curve gives a more detailed, leaf specific control.



Distribution for leaves, which grow only at branch ends

Colors: The From and To fields define a color variation range for leaves. Each leaf gets a randomly picked color from the defined range. The age distribution curve below these two fields define another color which depends on leaf's position in the plant (0 at the root, 1 at the end of branches). The final color is the product of the random color and the age dependent color.

Leaf colors variate from yellow to green. Old leaves are darker than young leaves.

Note: Leaf colors are evaluated and stored only when the Set Colors option of the plant's seed object is enabled. If colorization is done using materials and texture maps, which replace the colors computed using the options above, it is wise to turn the Set Colors option off to save some memory.

Level Object

A level object corresponds to 'directories' or 'folders' in Disk Operating systems.

Level objects allow all geometric objects in Realsoft 3D to be grouped into a hierarchical tree structure. For example, a table consists of two sub objects: a base and a cover. The base consists of any number of sub objects, such as legs and so on.

All geometric objects, including levels, have a 4x4 matrix associated with them. This matrix defines so called local object space for the object. All geometric attributes for the object are defined in this local space.

Linear transformations are applied to an object by combining a 4x4 transformation matrix to the object space matrix of the object. This transforms the geometry of the object, as well as the geometry of the sub objects. The 4x4 matrix is usually represented to the user as four vector fields: Translate, Scale, Rotate and Skew, but it can also be manipulated directly through scripts or the SDK.

The level object also plugs in a feature called 'Boolean Operations'. If the type of the level is 'Union' (OR) , then all the matter defined by the sub objects of the level is added into the universe. In other words, the logical operation between the sub object is OR.

If the type of a level is 'Intersection', then the sub objects are Boolean operated with each other using the AND logical operator.

Levels also allow geometric objects and material description objects to be grouped into a single entity. For example, one can create a wooden sphere by creating a level consisting of two sub objects: a sphere and a 'wood' mapping object. The level object property gadget includes the following controls:

- *Type of the level* - Defines how sub objects of the level are combined. The following alternatives are available:
 - *Boolean union (OR)* - The sub objects add matter to the scene i.e. the volume of the level is the sum of the sub volumes.
 - *Boolean Intersection (AND)* - The volume of the level is the intersection of the sub volumes.
 - *Boundary surface solid* - The surfaces of the sub objects are combined to define a volume in space. With this option, you can build a solid object from surfaces that do not define a natural volume themselves (a set of NURBS patches, SDS objects etc.). The surfaces should form a closed system without holes, otherwise rendering errors may appear.
 - *Material mapping average* - This option is useful for combining several materials into a weighted average. Most materials replace the result of the previous material: adding marble over a wooden sphere makes it look marble. With the mapping average level, you can make a weighted mixture of the materials, say 70 % marble, 30 % wood. The weight of each material can be defined using the 'Scope' value of the corresponding mapping object, or by assigning a scope value in each material.
- *Simulation* - Enables physical simulation computations for child objects in animations.

Link

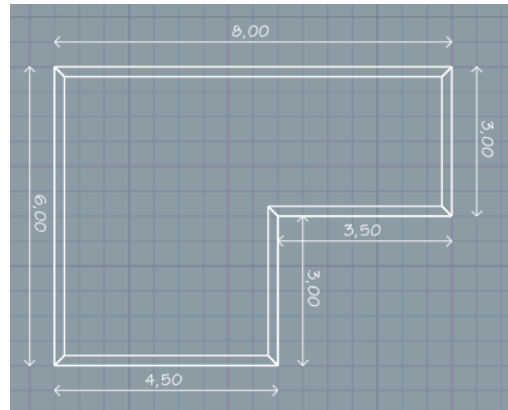
A link object serves only object evaluation purposes. You can use link pointing to a geometry object in systems which do not set special requirements for parameter objects. For example, you can use a link to define an animation path.

Measure

The measure object is displays its length in view windows. You can add measure objects to you scenes to show information about object dimensions.

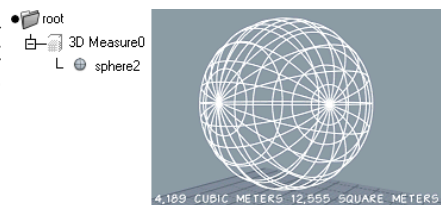
The properties of the measure object are:

- **Length** - The current length of the measure
- **Text Size** - The size of digits (in pixels), which display the length in view windows
- **Arrow Size** - The size of the arrow end symbols (in pixels)
- **Threshold** - Angle threshold for measure drawing. Measure lines, which are seen in a very low angle (viewing direction almost parallel to the measure line) should not be drawn, because the projected line becomes a single dot. This threshold controls when clipping occurs. Value 1 draws all lines regardless of the angle.



3D Measure

The 3D measure object computes and displays the volume and/or area defined by its sub objects. For the volume computation, sub object must be proper solids (analytic objects or properly closed boundary surface solids). Sub objects can include arbitrary boolean operations.



The properties of the measure object are:

- **Accuracy** - Accuracy of the computation. Minimal value 0 computes the volume and area of the bounding box. The higher the number, the more accurately the result will be computed. For example, the default accuracy 100 computes first 4 digits of the volume of a sphere correctly (error is less than 0.001 cubic meters when radius is 1 m).
- **Volume** - The current volume (a read-only property)
- **Measure Volume** - Enables volume measuring
- **Area** - The current surface area (a read-only property)
- **Measure Area** - Enables area measuring
- **Fused** - If set, overlapping volumes do not cumulate - they are added to the total sum only once. Similarly, surfaces traveling inside solid volumes do not contribute into the 'Fused' area. If the option is cleared, the result values are computed from each solid sub object separately and summed up.
- **Text Size** - The size of digits (in pixels), which display the volume in view windows
- **Show On View** - Displays the computed volume on view windows

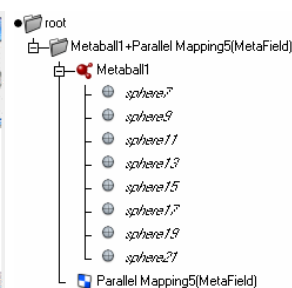
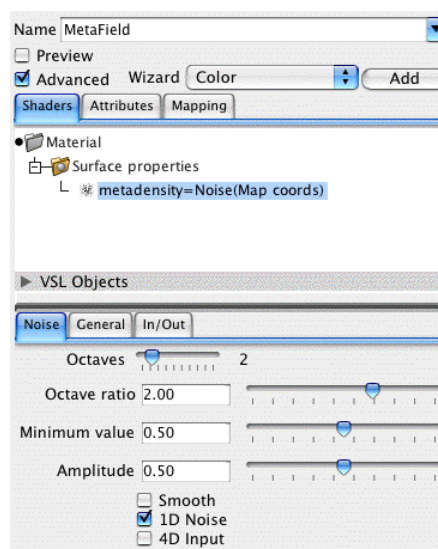
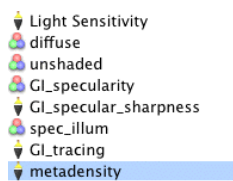
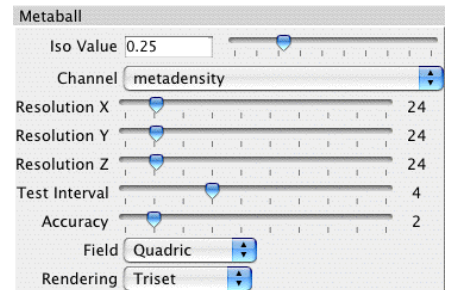
Metaball

The metaball level object creates a boundary surface for a density distribution defined by sub objects. The sub objects can be symmetric or elliptic spheres or 1D particles and a VSL material can be applied to modify the density in space. By default, the density decreases from 1.0 at half radius distance from the sphere center to zero at the sphere edge. The boundary surface of the metaball system follows the area where density is exactly a given Iso Value. Because the density distributions of metaballs melt together, the result surface behaves like liquid.

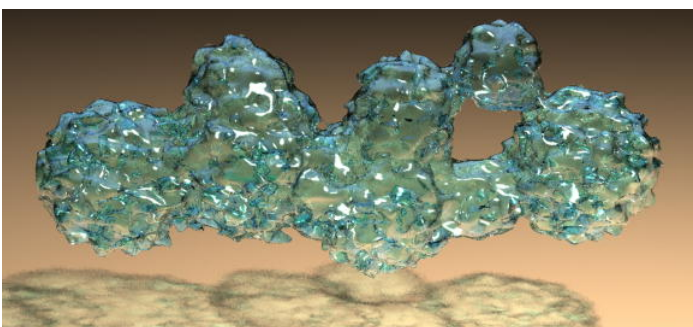
The following controls are available:

Iso Value: The density value which defines the surface. Decreasing the value makes the result object thinner.

Channel: An optional channel that modifies the density distribution. If set, the materials (more exactly, the Surface Geometry shaders) that affect the metaball system are evaluated. The total density field is the product of the spherical density fields and the material defined channel values.



Create a material that assigns a value to the density channel and map it to the metaball system.



Bumpy metaballs created by the system above. Because of the material defined density, only half a dozen metaball objects generate a detail rich surface

Resolution X, Y, Z: The accuracy of the 3D grid which is used for searching the iso surface. The resolution defines the density (and quality) of the generated surface. For example, if the diameter of a metaball system is 20 cm and the resolution values are 40, the surface will show details of the magnitude $20 \text{ cm} / 40 = 0.5 \text{ cm}$.

Test Interval: A speed optimization setting. Boundary surface is searched using the given test interval stepping along the Resolution X, Y & Z grid, and the exact surface is computed only if the density changes across the iso value limit. If the density distribution does not contain small details, the test interval can be several resolution units.

Accuracy: Defines how accurately the surface points are computed to match the exact iso value surface.

Field: Defines how the density field increases from zero at the metaball outer edges towards the full value 1. The higher the degree, the smoother metaballs melt together.

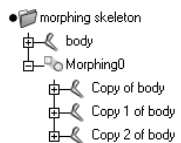
- *Linear* - Density increases at a constant rate along the sphere radius.
- *Quadric* - Density grows proportionally to the square of the radius.
- *Cubic* - Density increases proportionally to the third power of the radius ($d = c * r * r * r$).
- *Volume* - Density jumps to the full level at the metaball edge.
- *4th Degree* - The density decreases proportionally to the 4th power of the radius. The smoothest way to blend metaballs.

Rendering:

- *Triset* - the surface is rendered polygonally using the Marching Cubes Algorithm. This is usually the fastest way, but the result shape often suffers from various rendering errors which are typical to polygon models.
- *Procedural* - the surface is resolved during ray trace rendering without polygonization. This method renders a perfectly smooth surface, when the Accuracy setting is high enough. It is also extremely memory efficient.

Morphing

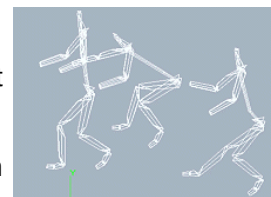
The morphing object is a level object which can contain two or more sub objects. The morphing object interpolates these sub objects and assigns the result to the target object. The target object must be at the same hierarchy level as the morphing object itself and the structure of the target object and the parameter objects must match.



The morphing object defines the following attributes:

- *Closed* - If set, the morphing generates closed interpolation where the first key object defines both start and end state.
- *Time values* - The knot line can be used for controlling the timing. By default, uniform timing is used, that is, the object distributes time evenly over the key objects.

Morphing object is derived from the Animation Object base class.



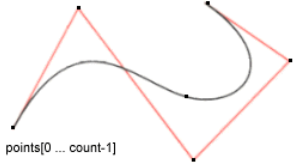
The target object 'body' controlled by a morphing object consisting of three key object

NURBS Curve

NURBS curves and surfaces are defined by number of four dimensional control points where the first three components define position in 3D space and the fourth component acts as a weight parameter. Increasing the weight pulls the curve towards the control polygon. Thanks to this property, NURBS curves can be used for representing quadric surfaces accurately.

Non-uniform parametrization allows representation of standard curves and surfaces. For example, one can use NURBS to represent standard Bezier and B-spline surfaces.

Realsoft 3D uses a very standard NURBS representation. Only the 'Closed' attribute needs some special attention for example when exporting Realsoft 3D NURBS objects to other applications. Many applications support only open NURBS curves and surfaces.



A NURBS curve is defined by the following attributes:

- **Control Points**
- **Order:** Order is degree + 1. For example, the order of a quadric curve is three. Order defines the minimum number of control points.
- **Closed** If true, the curve is closed. Realsoft 3D makes the curve periodic internally by extending the knot sequence and using a modulo operator.
- **Knots:** Knots define a set of parameter values in increasing order.

NURBS Curve Property Gadget

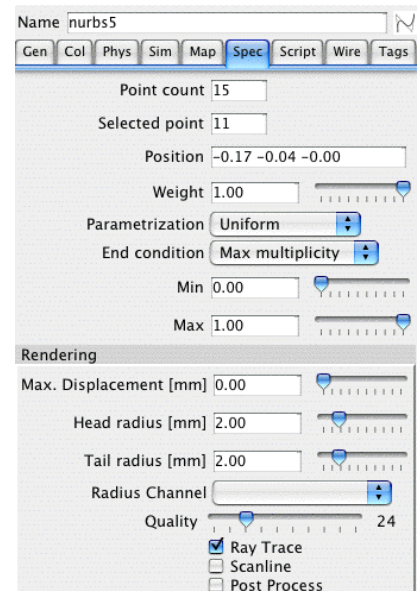
Point count: The amount of control points defining the curve. Reducing the point count usually changes the shape.

Selected point: The index (0, 1, 2 ...) of the currently selected point.

Position: The location of the selected point in 3D space.

Weight: The weight of the selected control point(s). Increasing the weight pulls the curve towards the selected control points.

Parametrization: Curve parametrization type. Parametrization affects curve shape significantly, and is relevant also in path animations. Uniform parametrization means that all point intervals represent an equal parameter interval. The chord length parametrization makes the parameter division using point distances.



End condition: Max Multiplicity end condition pulls an open curve to the first and the last control points. Phantom end condition uses a similar parametrization for the end points as for the interior points.

Min: The start parameter value for curve evaluation. With this and the next 'max' value, you can limit a suitable sub section from the curve.

Max: The end parameter value for curve evaluation.

This frame includes settings which control NURBS curve rendering in the Scanline module of the photorealistic render engine. The curve can be made visible in the final rendering using the Gen tab's visibility options.

Max. Displacement: The maximal amount of curve displacement defined either by moving the curve points or by changing the radius in a 'Scanline' shader.

Head radius [mm]: Thickness of the curve start (in millimeters).

Tail radius [mm]: Thickness of the curve end (in millimeters).

Radius Channel: An optional channel defining the curve diameter. If set, replaces the start and end radius values. The thickness values can be defined by creating a VSL material, where a scanline type shader assigns the values to the selected channel, and mapping the material to the curves. For example:

```
Material_for_Curves
  Scanline
    my_diameter_channel = Noise(map coords)
```

Quality: Rendering quality for curves. A smooth NURBS curve is drawn using short segments in the scanline or ray trace rendering. This value defines how densely the curve is subdivided to intervals. The higher the value, the smoother the result appears. The value is used per control point interval, so the same quality value works as well with a 5 point curve as with a 100 point curve.



Note

The NURBS curve supports also curve displacement by materials the same way as the thickness control presented above:

```
Material_for_Curves
  Scanline
    coordinates += Noise(map coords)
```

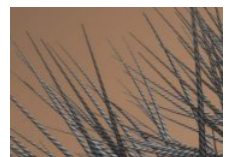
Ray Trace: If set, the curve is rendered using a ray traced geometry which casts shadows, is visible in reflections etc. The geometry type used in this context is quite memory efficient, and tens of thousands of curves can be included in the scene. The curve is procedurally refined towards a smooth shape during the rendering. The Quality value defines the computation accuracy. The minimal quality is usually sufficient for simple curves. High quality values may be necessary if a material is applied to deform the curve shape densely.



Hair created using ray traced NURBS curves

A ray traced curve defines the following geometry information that is available for VSL shaders:

- **UV Coords:** The first U component runs along the curve - head=0, tail=1. In all shaders except the Scanline V runs cylindrically around the curve, and the third component is the distance from the curve center. Scanline shader evaluates only the curve axis points, hence V & W are irrelevant in that context.



A spiral pattern mapped around curves using the native UV coordinates

- **U Direction, V Direction, Normal:** U direction is the tangential head-tail direction of the curve in the evaluated point. V direction is a direction orthogonal to U direction. In a Scanline shader, it defines the direction where the V component of UV coordinates is zero, and Normal channel is the cross product of the U and V directions. These directions are useful when displacing the curve position orthogonal from the curve tangent. In other shaders, where a real surface point is already examined, Normal is the usual surface normal and V direction is the cross product of U direction and Normal.

Scanline: If set, the curve is rendered using scanline geometry. This rendering method is often quicker than the ray trace approach, but scanline curves are invisible in reflections and do not cast ray traced shadows. The curve shape controls are the same as in ray tracing.

Post Process: Draws the curve in post processing phase, which is the last step in the rendering pipeline. Post processed curves do not react with lights or cast shadows, and do not show up in reflections. Due to the simplicity

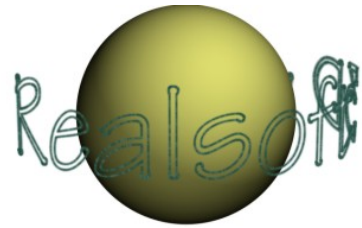
of the rendering method, they usually render quite fast. With this method, you can add for example text labels or rain effects into the image.

The curve shape can be controlled the same way using VSL materials as in ray tracing and scanline rendering. To define a color for a post processed curve, use a Post Particles shader. Color, Alpha and Fade channels defined in that shader control the rendering. Fade defines transparency.



Note

It is often impossible to add new post processed depth layers to an antialiased image from the ray tracer without getting some antialiasing errors. For example, a red sphere in front of a white background has some pink edge pixels, and rendering a post processed object between the red and white objects happens 'too late'. Such artifacts can be reduced by turning ray trace antialiasing off and by using image resolution down scaling after post processing.



Colored NURBS curves rendered in post processing

NURBS Surfaces

NURBS mesh is constructed from a two dimensional array of control points. Correspondingly, there are two sets of Count, Order, and Closed attributes: one set for each dimension.

NURBS Mesh Property Gadget

Attribute: Selects a channel for pointwise value control. The value can be set for the selected points using the value gadget below the channel selector.

Clear: Removes the pointwise values of the channel in the 'Attribute' field from all points.

Wrap U, Wrap V: Activates modulo wrapping into the unit interval for the selected channel component. This option is useful for UV coords channel and other texture parameter channels. Thanks to it, texture parameter can be interpolated correctly over the mapping seam on closed surfaces.

Count U: The amount of mesh control points in U direction. Reducing the point count usually changes the shape.

Count V: The amount of control points in V direction.

Selected: The index of the selected control point.

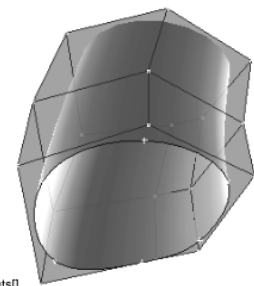
Position: The position of the selected control point.

Weight: The weight of the selected control point(s). Increasing the weight pulls the mesh towards the selected control points.

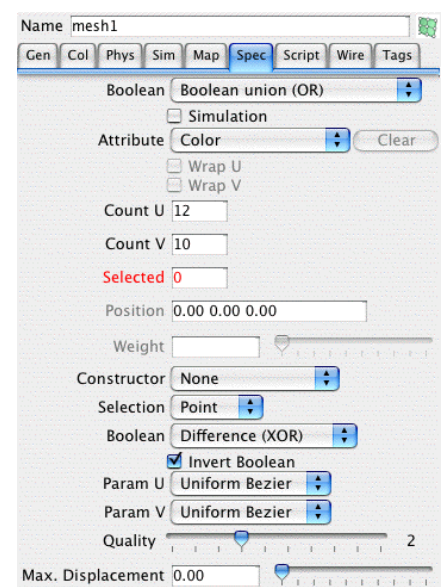
Constructor: Selects how child NURBS curves rebuild the mesh.

Selection: Selects geometry editing by single points or by complete row/column curves.

Boolean: The trim curve operation. This becomes relevant when the mesh has several trim curves. By default, the exclusive OR operation is used.



points[]



Invert Boolean: Inverts trimming. If set, trim curves cut holes. If not set, trim curves leave their interior left.

Param U, Param V: Parametrization affects mesh shape significantly, and is relevant also in path animations. Uniform parametrization means that all mesh patches represent an equal parameter square in parameter space. The chord length parametrization computes the parameter division from the patch size.

Quality: The quality in photorealistic rendering. NURBS mesh rendering is based on a numerical algorithm. The quality value controls the amount of algorithm iterations. The higher the value, the more accurately the 'ideal' surface shape is computed.

Max. Displacement: If the value of this field is greater than zero, and if surface geometry shaders are mapped to the surface, the surface is displaced according to the bump height channel in ray trace rendering. 'Max. Displacement' defines the maximal amount of displacement.

The displacement value is a real distance value (the unit can be selected from 'Options/Window' pull down menu, 'Metrics' tab), so usually quite small values are sufficient (such as 0.01 meters=1 centimeter). The amount of displacement has a great impact on rendering speed, so it is wise to use a minimal 'Max. Displacement' limit value here. Note also that displacement is symmetric to both sides of the original surface. Therefore, a displacement map that is centered around zero and whose amplitude varies between -1 and 1 centimeters renders faster than a bump map whose amplitude range is from 0 to 2 cm. In former case, the 'Max Displacement' value can be 1 cm, whereas the latter case requires a limit value of 2 cm.

Note: Use of displacement mapping does not automatically increase the rendering accuracy to meet the new surface detail requirements. If the displacement map includes lots of details, the original surface must be reasonably dense and the rendering Quality value described above must be high enough. If rendering artifacts appear, you can increase the mesh density using the CountU and CountV values described above.

Particle Objects

A particle object consists of a number of simple geometry items, such as points or lines. Despite of their simplicity, particles can create interesting rendering effects, thanks to the following reasons:

- A particle cloud or flock can include a very large number of items.
- Particle appearance can be customized using VSL materials.
- Particle effects can be mapped on particles.

Particles can be rendered as scanline objects before ray tracing or in the post processing phase after ray tracing. Scanline particles can be controlled using a VSL material with a Scanline shader. Post particle rendering can be controlled using a Post particles shader.



Note

All usual surface attributes such as color, transparency etc. affect appearance of scanline rendered particles. The scanline system passes the result of its rendering to the ray tracer, which in turn evaluates the surface attributes when combining the scanline data with ray traced objects. Therefore, you can even bump map scanline particles by modifying the surface normal in a surface properties shader!



Note

With a post particle shader, you can set and randomize many useful properties such as particle color, transparency, geometry points and diameters.

All particle types have the following controls on the property window:

Rendering tab

- **Post Processing** - If set, the particles are rendered in post processing phase of the rendering pipeline. This happens after ray tracing and is purely raster rendering, but distance and other channels can be taken into account. Post particles have only a limited interaction with ray traced items, but they are very quick to render. A Particle effect must be mapped to the particles, otherwise they does not render anything visible.
- **Scan Line** - If set, particles are rendered by the scanline system before ray tracing, meaning that transparent particles show the items behind them, and reflective particles can mirror ray traced objects.
- **Ray Tracing** - If set, particles are rendered as ray traced objects. This is usually the most memory intensive and slowest solution, but gives the most realistic results. 1D particles are rendered as spheres, 2D particles as cylinders or cones with rounded ends and 3D particles as elliptic spheres.

Geometry tab

- **Count** - The number of points in the particle object.
- **Properties** - A list of attributes that can be assigned as a shared value common to all points or as a pointwise value.
 - *Speed* - Defines how fast particles move. The value is used when the Emitter construction is active.
 - *Life Time* - Defines how long particles live (in seconds). When a particle dies, a new one is created to replace it. Therefore, the Life Time attribute has meaning only together with the emitter construction.
 - *Color* - The usual diffuse color of particles.
 - *Velocity* - The motion vector of particles. Defines how much particle(s) move in one second.
 - *Transparency* - Transparency property for rendering.
 - *Reflection* - Reflectivity for rendering.
 - *Alpha* - The alpha channel value.
 - *Fade* - The Fade channel makes scanline particles translucent. Also some particle effects use the fade value for transparency control.
 - *Illumination* - Self illumination color. Used in scanline rendering.
 - *Optical Thickness* - A well known but in particle rendering only rarely useful optical property.
 - *Scope* - A material mapping attribute.
 - *UVW Coordinates* - Defines the position of each particle in the 'Emitter' and 'Surface' construction methods. This property is initialized to a random value when a construction is activated first time.
 - *Other user channels*
 - - All custom channels of the Channels library are available here for rendering control purposes.
- **Value** - The value of the currently selected attribute. If 'pointwise' is set, the gadget shows the value of the selected points, otherwise all points share the same common value.
- **Render Channel** - The rendering engine channel to which a pointwise attribute is copied at rendering preprocessing time. This way all properties can be used to control particle shading.
- **Pointwise** - Makes an attribute pointwise i.e. a value per particle point is allocated. Every particle can have its own private value for the attribute.

Note: A pointwise value for all rendering channels (Color, Fade etc.) can be initialized using the map2Obj tool.

Constructor tab

- **None** - No particle construction applied. Particles can be freely moved around in 3D space.
- **Distribute over surface**- The particles are placed randomly around the surface of the sub object. To use many objects, collect them into a single level.
- **Emitter** - Particles are generated on the surface of the sub object with defined life time and velocity properties. New particles appear as old particles die. The following attributes control the behavior of emitted particles:
 - *Life time randomness* - Defines how much the life time of created particles varies. Value 1 will produce a totally random life time between zero and the object's common Life Time attribute.
 - *Speed randomness* - Randomizes the speed of emitted particles. If the value is zero, all particles have the same speed (defined by the common Speed attribute of the object).
 - *Direction randomness*- Randomizes the direction to which emitted particles travel. Value 1 will produce a totally random direction. Value zero will emit the particle to the exact surface normal direction of the creation point.

In addition to these three controls, also the common/pointwise properties of particles defined in the Geometry tab control the particle behavior. Common values define the range for randomization and pointwise values are used until the particle dies and a new one is created to replace it.

1D Particles

A 1D particle object consists of a set of points. Points are suitable for modeling dust clouds, symmetric flares, stars and other objects which do not need a 'direction' property.



A moving sphere emitting 1D scanline particles

1D particle specific properties:

- A 1D particle renders in scanline as a round disk that is facing towards the camera.
- Scanline particle location can be displaced using a 'Scanline shader' in a VSL material. The 'Coordinates' channel defines the position.
- 1D scanline particle object defines the following UV space (UV Coords channel): U runs from 0 at the middle point of a particle to 1 at the edge of the particle. V runs from zero at the first particle to 1 at the last particle.
- The 'Size' property at the geometry tab defines the radius of particle discs.
- In post processing, 1D particle drawing can be controlled by using the following channels in a 'Post particles' shader:
 - *Color* - particle color
 - *Fade* - particle transparency
 - *Coordinates* - position of a particle
 - *V direction.X* - the first sub channel defines particle radius

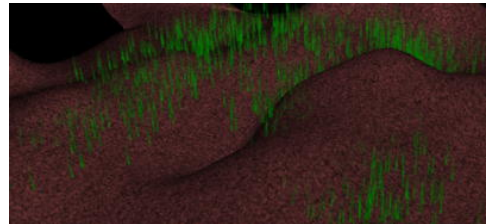
- Ray traced 1D particles are rendered as spheres, unless the **Ray trace as quadrangle** option is set, in which case each particle is rendered as a flat square facing towards the camera. Quadrangles define also UV coordinates, which makes texturing easy.

2D Particles

A 2D particle object consists of a set of straight lines. Lines are suitable for modeling rays, fur etc.

2D particle specific properties:

- Particle ends can be displaced using a 'Scanline shader' in a VSL material.
- 2D particle object defines the following UVW space (UV Coords channel) in scanline drawing: U runs from 0 at the start point of a particle line to 1 at the end of the particle. V runs from zero at the center line of a particle to 1 at the edge of the particle. W runs from 0 at the first particle to 1 at the last particle.
- 2d particles have the following additional properties:
 - *Diameter1* - The diameter of the start point of the particle in millimeters.
 - *Diameter2* - The diameter of the end point of the particle in millimeters.
 - *Length* - The length of particles (in the current measuring unit).
- In post processing, 2D particle drawing can be controlled by using the following channels in a 'Post particles' shader:
 - *Color* - particle color
 - *Fade* - particle transparency
 - *Coordinates* - start point of a particle
 - *U direction* - end point of a particle
 - *V direction X and Y* - two first sub channels define the radius *2D particles sprayed onto a surface* values at the end points



3D Particles

3D particles define a full 3D coordinate system. They are therefore suitable for effects that require full orientation control in 3D space. The current post effect set does not require such control.

- 3D particles can be ray trace rendered as as spheres/ellipsoids or as rectangular planes. The latter alternative, available via the **Ray Trace Rectangles** option, is a very suitable feature, when a large number of UV textured rectangles is needed. 3D particle rectangles are rendered in a memory efficient manner and they define UV coordinates. You can easily texture them to represent tree leaves etc.

The post processing control channels of 3D particles, available in a 'Post particles' shader, are:

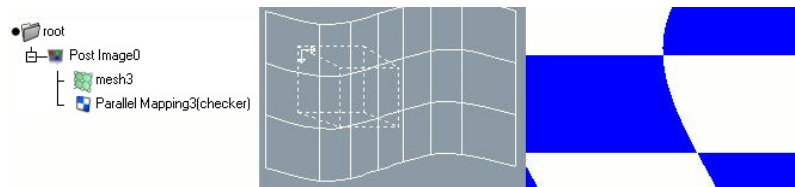
- *Color* - particle color
- *Fade* - particle transparency
- *Coordinates* - the middle point of a particle
- *U direction* - a control point defining the first elliptic radius

- *V direction* - a control point defining the second elliptic radius
- *Normal* - a control point defining the third elliptic radius

Post Image

The Post Image is a special kind of particle object. It is classified as a particle object because the 'Render particles' post effect module renders it at the post processing phase. It is a level object, which can contain another geometry object. The sub object can be a rectangle, a NURBS mesh or another 2D geometry.

The geometry defines an image plane warping. Each point of the rendered image corresponds a point in the surface. The image plane can be colored by using a material which includes a 'Post particles' shader. By animating the material mapping or the sub geometry, you can create animated waves, warps and other interesting effects.



A post image including a NURBS mesh mapped with a checker pattern. A part of the result image on the right.

Path Level

The Path Level object is used - as the name suggests - for path animations. It uses two sub objects: the actual motion path and an axis curve, which defines the horizon plane for banking control.

Path Level implements some useful banking behaviors and an option for constant speed motion. These are not available when using a plain NURBS curve as a motion path.

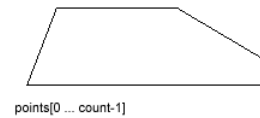
Path Level has the following controls in its property gadget:

- Path rotation type - one of the following banking options:
 - No banking - All banking with respect to the horizon is eliminated.
 - Natural banking - The target is rotated to match the normal of the curve in a non-cumulative manner.
 - Cumulative banking - The target is rotated to follow the curve using the shortest rotation route between each two consecutive positions. This usually banks the target i.e. rotates it around the curve. Banking cumulates towards the end of the path.
 - Aeroplane banking - The banking behavior simulates aeroplane. Banking increases when the path turns and returns back to zero during straight sections of the path. The length of the horizon axis under the path level controls the amount of banking. Making the axis longer decreases banking.
 - No rotation - The targets are not rotated at all.
- Constant Speed - selects point density independent motion.

Polygon

An analytical polygon is basically a trimmed plane. Its geometry is defined by three or more geometry points on a plane.

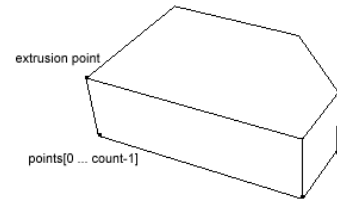
Because polygons are planar, its volume is a half space. In other words, the one side of the polygon is said to be inside the polygon and the other side the exterior. For example, you can use the polygon object to split an object into two parts using the boolean operations.



Polyhedron

A polyhedron is derived from a polygon by introducing so called 'extrusion point'. The extrusion vector can be computed by subtracting the first geometry point from the extrusion point. The top and base planes are perpendicular.

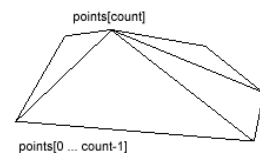
The polyhedron supports fast Boolean operations.



Polymid

A polymid object is derived from a polygon by introducing a new 'Apex' attribute.

The polymid supports fast Boolean operations.

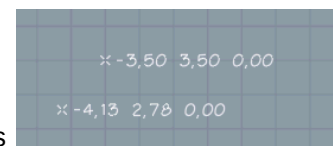


Position Mark

The position mark object displays the coordinates of its location in view windows. You can add position marks to your scenes to show information about important positions.

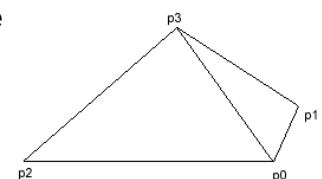
The properties of the position mark object are:

- **Position** - The XYZ coordinates of the object
- **Text Size** - The size of numerical digits (in pixels), which display the coordinates in view windows



Pyramid

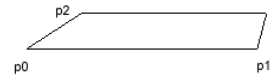
An analytical pyramid is defined by four points: three of them define the base rectangle and the fourth point defines the apex.



Rectangle

An analytical rectangle is defined by three points. Opposite edges are always parallel to each other. The fourth corner point is computed from the given three points internally.

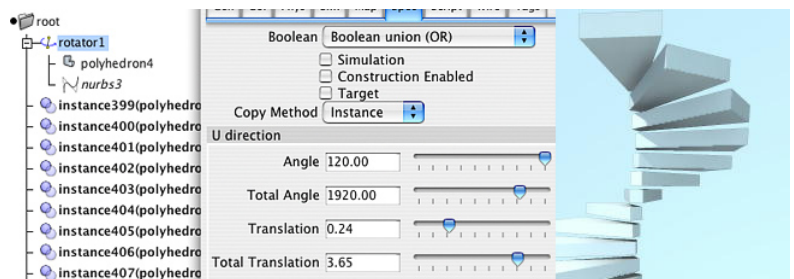
The volume defined by a rectangle is an infinite half space, trimming out the other side of the plane. Which one of the sides is 'inside' depends on the winding order of the points. The interior side is defined by the surface normal computed using the right hand winding rule.



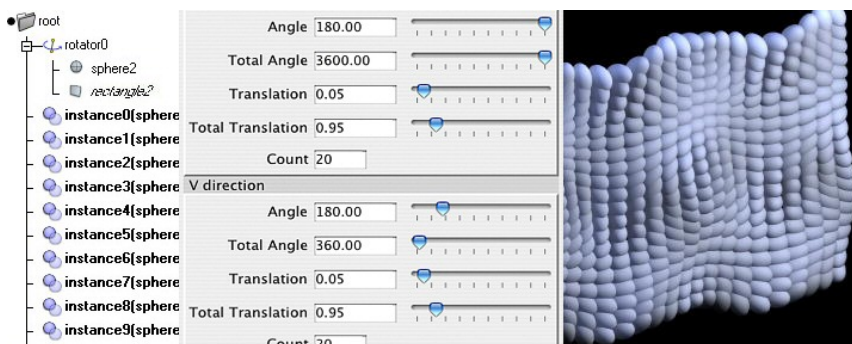
For example, one can use a rectangle object to split an object into two parts using the Boolean operations.

Rotator

A rotator object produces duplicates or instances of its first sub object by rotating and/or translating the copies into regular rows and circles. If the second parameter object is one dimensional (a NURBS curve), the result is a sequence of objects. If the second sub object is two dimensional mesh, the result is an array of objects.



Spiral stairs created by a 1D rotator system



A sphere wave created by a 2D rotator system

The following set of controls is available. The 'U direction' controls are applied for all kind of parameters. The V direction controls become relevant when the 2nd parameter is two dimensional. Both sets of controls work in a similar way.

Copy Method: The object duplicates can be either real objects or instances. Instancing saves some memory especially when the sample object is complex.

Angle: Defines how much a new duplicate is rotated from the previous duplicate around the current parameter object evaluation position.

Total Angle: The same as the previous one, but expressed as the total rotation for all duplicates.

Translation: Defines how much a new duplicate is moved along the 2nd parameter U/V dimension. The amount is measured in parameter space i.e. value 1 corresponds to full parameter length.

Total Translation: Amount of total translation from the sample object to the last copy/instance.

Count: Controls the amount of resulting objects (or object rows/columns in 2D case).

Roof

The roof object creates a standard roof shape from its parameter curves: a roof edge polyline and the optional roof top polylines. If no roof top lines are defined, the object computes a straight line across the roof, which splits the roof symmetrically.

Roof object can operate as an individual object, or in conjunction with a building object. When placed inside the hierarchy of the building object, the building object automatically stores a roof edge line computed from the walls into the roof object.

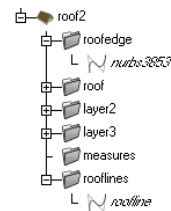


Note

Roof object's Build option can be disabled and the geometry can be replaced with any user modeled roof shape. Wall object's Extend to Roof will then fit the walls with the custom roof shape.

The hierarchy of the roof object contains the following sub levels:

- **roofedge:** Contains the polyline, which defines the roof edge.
- **roof:** contains the auto generated top surface (coating) of the roof (a polygonal SDS mesh). You can place material attributes into this folder. Editing the geometry while the roof's Build option is enabled makes no sense, because next construction refresh will remove editing.
- **layer2:** the second roof geometry layer (support below the coating).
- **layer3:** third roof geometry layer (the actual body of the roof with insulation etc).
- **measures:** the measure lines showing the dimensions of the building are stored here, when the respective option is turned on.
- **rooflines:** the Roof Line tool places the polylines, which define the roof top, to this folder. Roof lines can be created also manually using the NURBS curve tool.



The roof object has the following controls in the Property window's Spec tab:

Roof Geometry options:

- **Build:** enables automatic construction. Generated roof geometry can be edited manually only after Build is turned off. Turning Build on removes manual changes. When the option is enabled, only the parameter data is saved to disk files, which usually makes the file size small.
- **Auto Texture:** if enabled, the roof object automatically adds and updates material mapping objects for walls, ceiling, floor and base.
- **Roof End:** Select from three alternatives, Straight (gable roof), Match Roofline (user defined angle) and Beveled (hipped roof).
- **Height:** Basic height of the roof. Note that when roof lines are included, the lines define the height. Move the roof lines to change the height.
- **Eave Width:** horizontal distance from the wall to the roof edge.
- **Eave Elevation:** Roof height is measured starting from the base height of walls. Eave Elevation moves the actual roof edge up or down from the zero level.



From the left: Straight, Beveled and Match Roof Line options

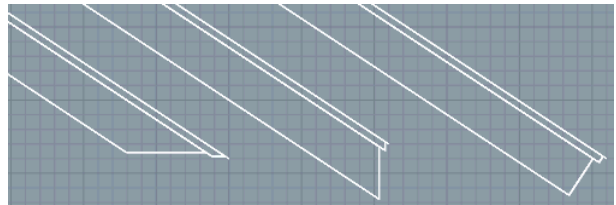
Sub Layer 1 and 2 have identical options:

- Add Layer: the respective roof layer geometry is added to the scene.
- Filled Top: build a surface between the roof layer and the layer above it. Usually the layer is invisible inside the roof structure, so it can be left out from the model just to make rendering a bit faster.
- Bottom Fill: Defines how the roof layer is filled from below. Perpendicular fill adds a surface which is parallel to the roof top. Horizontal option fills the bottom with a flat horizontal surface.



Parallel and Horizontal roof layer fill

- Roof Edge: Can be Horizontal, Vertical or Perpendicular. See the picture on the right.
- Thickness: the thickness of this roof layer.
- Offset: Distance from the edge of the upper layer to this layer.



From the left: Horizontal, Vertical and Perpendicular roof edges

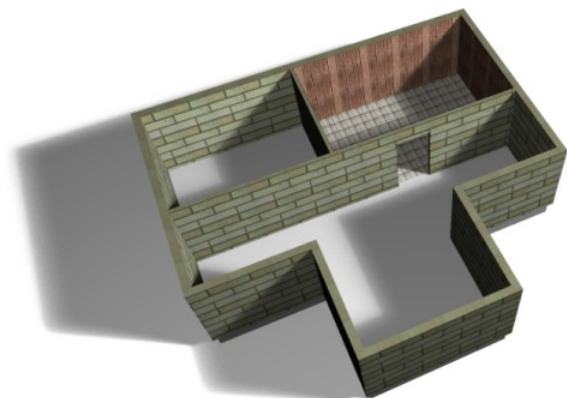
Measure Lines:

- Measure Lines: Enables measure line drawing.
- Offset: the distance from the wall polylines to the measure lines.
- Text Size: defines the font size (in pixels) for the displayed dimension values.
- Arrow Size: size of the end arrows (in pixels)

Room

The room object creates new surface layers around the surrounding walls, roof and ceiling. The room surfaces can be decorated independently from the rest of the house. The object can also generate automatically mouldings for the floor-wall edges and the wall-ceiling edges.

The surfaces are extracted automatically from the parent building object. The position indicator of the room object defines, which room space will be decorated. Moving the indicator to a new room moves the decorations as well. A room object moved outside the hierarchy of the building is inactive.



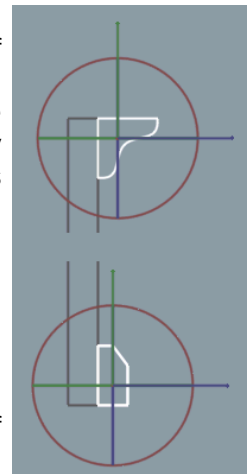
One room object added to the building

The room object has two special sub folders called *floormouldingsample* and *ceilingmouldingsample*. When an example section of a moulding is placed in either of these folders, the object automatically duplicates, stretches and transforms the moulding to travel around the respective floor/ceiling edge of the entire room. The example moulding can be placed anywhere in 3D space. Its object space must be properly oriented: the green y axis must point upwards and the blue z axis must point outwards from the wall side of the moulding.



Note

Repeating moulding structures, such as wooden panel walls, can be defined using a replicator object.



Object spaces for floor and ceiling mouldings

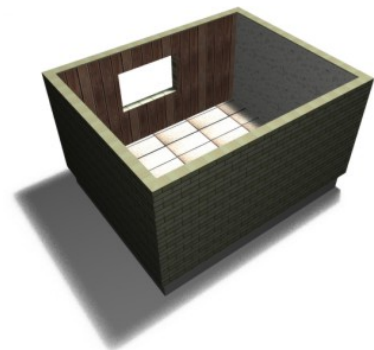
The Property window's Spec tab controls for the room object are listed below. Some of these options are available in the tool bar, too.

Room Options:

- **Build:** Enables room construction. Generated surfaces can be edited manually only after Build is turned off. Turning Build on removes manual changes. When the option is enabled, only the parameter data is saved to disk files, which usually makes the file size small.
- **Texture Floor:** Adds a floor surface to the room.
- **Texture Ceiling:** Adds a ceiling surface to the room.

Walls:

- **Texture Walls:** Enables wall decoration. When set wallpaper surfaces are created.
- **Wall Range, From, To:** When Wall Range is set, only the walls with indices between From and To are decorated. Indexing starts from the wall closest to the indicator.



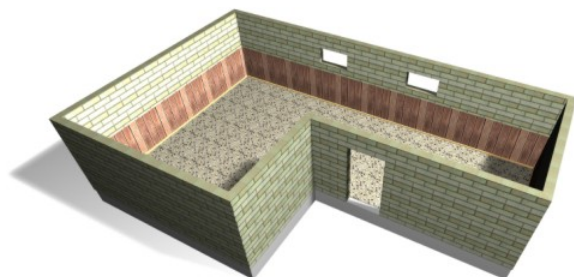
Note

It is possible to add several room objects to a same room space, in order to apply several different textures to the walls. Only one room object should have the roof and floor surface creation enabled.

Two room objects applied to the same space

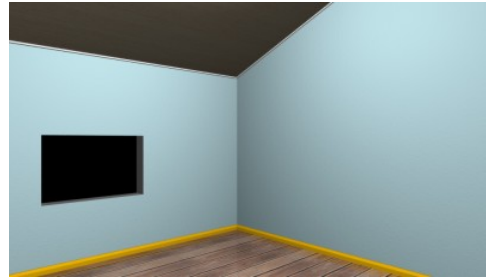
Wall Height:

- **Start, End:** The relative height of the textured area of the walls.
- **Extend to Roof:** Enables automatic wall surface extension to the limiting roof surface. This option can be useful, when the room space is not limited by a flat ceiling, such as attic rooms.



Start=0 and End=0.5 textures lower half

- **Wall-Roof Penetration:** A tolerance factor, which helps to close the gap between extended wall surfaces and the roof.
- **Fitting Interval:** The wall surface height is matched with the roof at every corner and where a roof line crosses the wall. If the roof has a complex or even a smoothly bending shape, a low fitting interval (such as 0.1 meters) may be necessary. This option defines a maximal distance between two fitting positions.



emphasis>A room in the attic

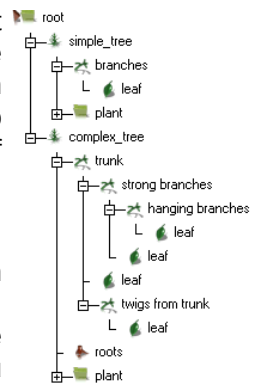
Measure Lines:

- **Draw Measure Lines:** Enables measure line drawing. The lines are always drawn inside the room.
- **Offset:** The distance from the wall polylines to the measure lines.
- **Text Size:** Defines the font size (in pixels) for the displayed dimension values.
- **Arrow Size:** Size of the end arrows (in pixels)

Seed Object

Plant seed object is the highest hierarchy level of a plant object. It builds a fractal plant using growth rules defined by its attributes and child objects (branches and leaves). The seed object includes a 2 point line geometry, which defines the initial growth direction and speed. All dimensions of the plant (branch diameters, leaf sizes etc.) are relative to the length of the line. Plants have some gravity related attributes, and the direction of the z axis (the blue object handle) defines the direction of the gravity.

The seed object generates the geometry of the plant under its own hierarchy. You can assign materials and constant attributes (color etc.) to the created plant hierarchy levels. Do not change the actual geometry items, because they are regenerated every time when a growth parameter changes and when loading a plant object from a file. If you need to edit the geometry items, move them away from the plant hierarchy.



Plant hierarchies. The simple hierarchy on the top grows the roots and the branches using the same rules. More complex trees use sub objects to define how trunk and branches grow.

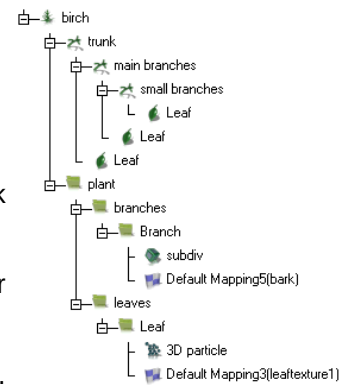
To create a plant seed object:

- Activate the Plant tool from the tool bar or from the pull down menu.
- Click on a view window to define the point where the plant grows.
- Move the mouse upwards to define the direction and speed of the growth. Click second time to finish the tool.

The plant tool automatically inserts a branch object into the seed object in order to set up a fully functional plant.

To change the direction of growth from the root, point edit the plant's axis line.

Do not rotate the tree, because that would rotate the gravity direction as well. *Materials assigned to the tree*
It is probably a good idea to make the generated plant geometry invisible while *geometry* point editing the axis line.



Plant seed object includes the attributes described below.

Height: Controls the height of the plant. The value defines the approximate length of growth during the first year. The value is not exact because Random Length and other branch attributes contribute into the result.

Thickness: The thickness of the plant. The value defines the initial radius of the plant in the point where growing starts. Note that the branch profile and other attributes may change the radius from the value defined here.

Random Seed: Seed value for the computation of randomized plant properties. Change this to get unique trees.

Render Plus: Adds plant complexity for photorealistic rendering. The idea is to use easily manageable preview objects in the modeling stage and render the full detail only in the final images. The defined value is added to the plant's age, and the leaf density parameters are multiplied with (1+value). Because the number of branches usually grows exponentially by age, plant's complexity grows very rapidly at every render plus step. Usually a small value like 1 is enough; it typically increases the complexity by a factor of 8.

Max Complexity: Limits plant's complexity (number of branch segments + number of leaves) to a safe value range. Some attributes increase complexity exponentially. Mistyping a high value could rapidly consume all memory, cause heavy memory swapping and make it hard to recover from the error. The complexity limit protects you from this kind of troubles. The default value of complexity is a plant root's class attribute; use the preferences window to adjust it to match the performance of your computer system permanently.



Modeling vs. rendering complexity

Set UV: If set, the following information is stored into the tree geometry's UV Coords channel:

- UV Coords.X - 0..1 parameter around the branch
- UV Coords.Y - 0..1 parameter, which starts from zero at the root ends, and reaches 1 at the end of branches.
- UV Coords.Z - the ratio (current branch radius/radius at the root)

A simple color gradient material

Set Colors: If set, the plant computes and stores colors for leaves and other items which include colorization features. Alternatively, the plant can be textured using materials in the plant hierarchy.

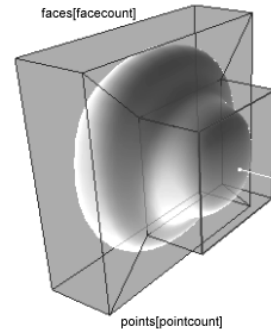
Subdivision Surfaces

Realsoft 3D's Subdivision Surface implementation is based on Catmull-Clark surfaces. However, Realsoft 3D supports rational SDS surfaces - there can be a 'weight' parameter associated with each control point. Increasing the weight pulls the corresponding surface point towards the control point in question.

A subdivision surface object is defined by a set of control points and a set of faces. Each face can refer to three or more control points.

There are many pointwise attributes that can be defined, such as color, illumination, or transparency. Also, there can be pointwise values for any user defined channel.

Vertex sharpness can be set for any set of vertices. It can be Rounded (default), Semi Sharp or Sharp. There's also a similar control for edge sharpness. Additionally, a 'Free' edge sharpness rounding can be defined numerically.



SDS objects can include multiple UV channels defining how textures are applied on the surface.

Realsoft 3D's SDS implementation also supports per face materials: a material name and a set of UV parameter values (one per vertex in the face) can be attached to any face.

SDS Properties

Handles

SDS supports a small but very powerful set of handles through which most commonly needed tools can be applied.

There are three sets of handles: point, edge and face handles, which can all be selected. When a handle is selected, it typically shows a set of sub handles. For example, when a face handle is selected, also the face normal handle is shown for the selected face.

The usual selection modifiers and the drag box selector are available when selecting SDS handles.

Point Handles

The point handles are shown when the SDS object is in the point editing mode.

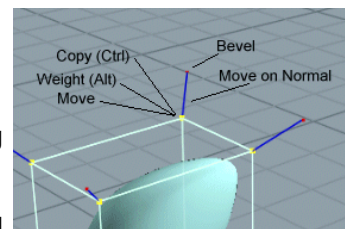
Each SDS point acts as a translation handle by default.

Points can be duplicated by dragging them with the Ctrl key pressed.

The weight parameter associated with the points can be changed by dragging the point with the Alt key pressed.

The 'Move on normal' tool can be activated by dragging the surface normal handle.

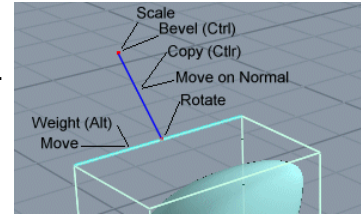
The Bevel tool can be activated by dragging the end point of the surface normal handle.



Handles shown in the point editing mode

Edge Handles

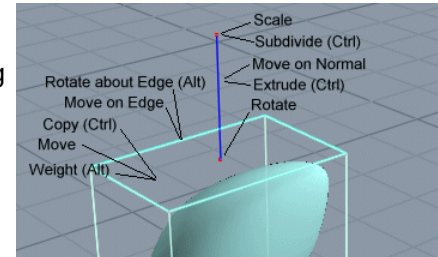
The edge handles are shown when the SDS object is in the edge editing mode.



Handles shown in the edge editing mode

Face Handles

The face handles are shown when the SDS object is in the face editing mode.



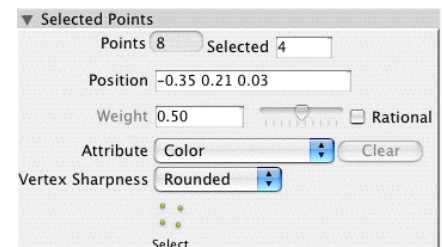
Handles shown in the face editing mode

Property Gadget

The SDS property gadget has the following frames: SelectedPoints, SelectedEdges, Rendering, MultiUV, Faces.

Selected Points frame

- Points - the number of vertices in the SDS object
- Selected - index of the selected vertex. If more than one vertex is selected the index of the first selected vertex is shown in the conflict color.
- Position - 3D position of the selected vertex.



- UV - current UV channel value for the selected vertex. The current UV channel can be selected from the list in the Multi UV frame.
- Attribute - list of pointwise (defined separately for each vertex) attributes. Use this to select the pointwise attribute.
- Value - selected pointwise attribute value for the selected vertex.
- Clear - removes the selected pointwise attribute from the SDS object.
- Rational - if checked pointwise weights can be controlled. Unchecking this checkbox removes pointwise weights.
- Weight - sets the weight for the selected vertex/vertices. Can be used only when the Rational checkbox is checked.
- Vertex sharpness - sharpness of the selected vertices. Possible values are
 - Rounded
 - Semi Sharp
 - Sharp

- Select - selection mode, select by Point, Edge or Face.

Selected Edges frame

- Edge Sharpness - sharpness of the selected edges. Possible values are

- Rounded
- SemiSharp
- Sharp
- Free



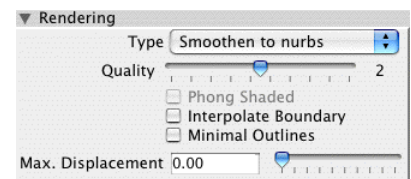
If free is selected, then the edge rounding can be set numerically.

- Edge Rounding - can be set to edges that have Edge Sharpness set to Free.

Rendering Frame

- Type - How the SDS object will be rendered. The possible values are

- Smoothen to nurbs - the surface is rendered using bicubic NURBS patches
- Smoothen to rectangles - the surface is rendered using bilinear phong shaded quadrangles
- Smoothen to triangles - the surface is tessellated to phong shaded triangles



- Polygonal - The original control polygon is rendered without tessellation. Normal smoothing can be turned on using the Phong shaded option described below.
- Quality - Rendering quality, the higher the value, the smoother the surface. Controls how many NURBS patches are fitted into the surface.
- Phong Shaded - If set, normals of a polygonally rendered SDS surface are interpolated using Phong shading making the surface appear smoothly curved. This method works well only with relatively dense control polygons.
- Interpolate Boundary - If set sets maximum crease value for boundary edges (i.e. edges that are not shared by two faces).
- Minimal Outlines - If set, the density of outlines appearing on the rendered surface in outline rendering mode is minimal (i.e. matches the density of the control polygon). If unset, NURBS patch fitting density defined by the Rendering Quality setting above and triangularization of polygonal surfaces are visible in rendered outlines.
- Max. Displacement - If the value of this field is greater than zero, and if surface geometry shaders are mapped to the surface, the surface is displaced according to the bump height channel in ray trace rendering. 'Max. Displacement' defines the maximal amount of displacement. The displacement value is a real distance value (the unit can be selected from 'Options/Window' pull down menu, 'Metrics' tab), so usually quite small values are sufficient (such as 0.01 meters=1 centimeter). The amount of displacement has a great impact on rendering speed, so it is wise to use a minimal 'Max. Displacement' limit value here. Note also that displacement is symmetric to both sides of the original surface. Therefore, a displacement map that is centered around zero and whose amplitude varies between -1 and 1 centimeters renders faster than a bump map whose amplitude range is from 0 to 2 cm. In former case, the 'Max Displacement' value can be 1 cm, whereas the latter case requires a limit value of 2 cm.

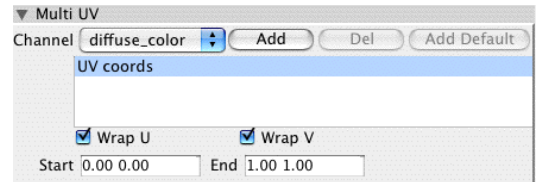


Note

Use of displacement mapping does not automatically increase the rendering accuracy to meet the new surface detail requirements. If the displacement map includes lots of details, the original surface must be reasonably dense and the rendering Quality value described above must be high enough. If rendering artifacts appear, you can apply the SDS Smooth tool to increase the control polygon density.

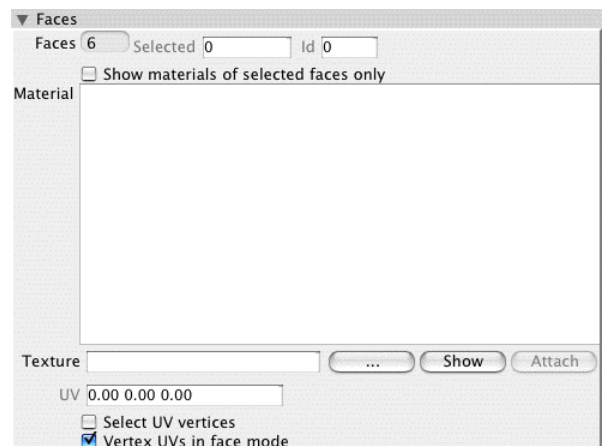
Multi UV frame

- Channel - possible UV channels.
- Add - add the selected channel as a UV channel.
- Del - Delete the selected UV channel. The selection is shown in the list view gadget.
- Add Default - adds the default UV channel (UV coords).
- List of UV Channels added to the object.
- Wrap U - If checked interpolates U to nearest modulo at the boundaries defined by Wrap Start and Wrap End. Check this if the UV space is closed in the U direction.
- Wrap V - If checked interpolates V to nearest modulo at the boundaries defined by Wrap Start and Wrap End. Check this if the UV space is closed in the V direction.
- Wrap Start - Minimum boundaries (U, V) for modulo wrapping. This should usually be 0.0 0.0 for texture maps.
- Wrap End - Maximum boundaries (U, V) for modulo wrapping. This should usually be 1.0 1.0 for texture maps.



Faces frame

- Faces - number of faces in the SDS object.
- Selected Face - index of the selected face. If more than one face is selected the index of the first selected face is shown in the conflict color.
- Selected faces only - if checked the Material list view below shows only those face materials and face textures that are attached to the selected faces. Otherwise all face materials and face textures are listed.



- Material - shows the face material and face texture names of the selected faces or of all faces. This control has a context menu where the selected face material or face texture can be detached from the faces. If Selected faces only is checked then the face material or face texture is detached from the selected faces. Otherwise it is detached from all the faces.
- Texture - Select a image file to be used as a face texture for the selected faces. Click the Attach button to attach the face texture.
- Attach - Attach the selected image file to the selected faces as a face texture.

- UV - the face material or face texture UV for the selected UV vertex. Check the Select UV vertices checkbox and select a UV vertex in the UV Editor window. Now the UV control can be used to set the face material or face texture UV for that vertex numerically.
- Select UV vertices - if checked UV vertices can be selected in the UV Editor. Otherwise faces can be selected.
- Vertex UVs in face mode - if checked per vertex UV values are shown in the UV Editor when the SDS object is in the Face selection mode. Otherwise face material and face texture UVs are shown.

Additional Controls in The Wire Tab

- Material Rendering - Defines how the SDS object is rendered in shaded mode.
 - Single - Only one face material or face texture per face is rendered. This is much faster than the Full mode.
 - Full - All face materials and face textures as well as mapping objects are considered. It is often preferable to use low Texture Quality settings when using this mode.
- Force Tessellation - If set, triangular tessellation of faces which have more than 4 vertices is shown in wireframe drawing.

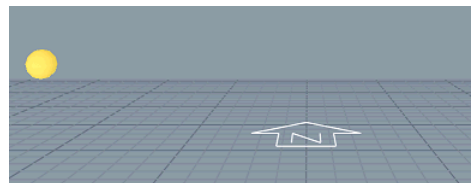
SunMoon Object



SunMoon object manages its first child object to simulate the position of the sun or the moon in a given date and a location.

The date is defined using local time. The applied time zone correction is geographical; the computation may include one hour error at certain time zones. The daylight saving time is not considered.

The location is defined by the usual longitude and latitude coordinates. Both coordinates should be given in decimal degree form (not as degrees.minutes.seconds).



Sun's position at Tampere, Finland 21 o'clock in the midsummer evening

To simulate the position of the sun or the moon, the sunmoon object rotates its first child object by its pivot point. This means that the azimuth and altitude angles of the selected target (sun / moon) can be found from the Rotate values of the object space of the first child. The first component, the heading angle, is the azimuth and the second component, the pitching angle, shows the altitude. Zero heading corresponds to the geographic north (= negative z axis of the sunmoon object space).

A sunmoon object can be created using the Sunmoon Tool. Some examples:

Point light sun

1. Create a point light, which represents the sun. You can add some visible objects like a glowing, shadow invisible sphere around the point light. Make sure all components of the sun are collected under a single hierarchy level. If not, select all components and apply Drop to level from the popup menu of the select window.



Note

The position of the sun does not matter. However, its distance from the origin should be such that the visual size of the modeled sun appears correct when viewed from the origin. The distance should also be large enough (e.g 1000 times the size of the rest of the scene). Otherwise the position of the sun might change when the camera moves around the scene. However, do not try to use the real astronomical scale; the accuracy of computers is not sufficient for shadow computations from a light source that is millions of kilometers away from the illuminated objects.

2. Select the sun and click the SunMoon tool. It is located in the Light Sources tool tab.
3. Adjust the date and the location using the toolbar controls.

Distant light sun

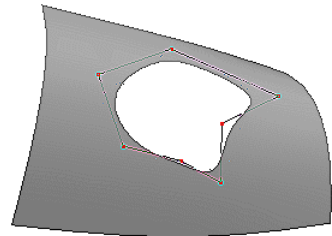
1. Create a distant light, which represents the sun. The position of the light does not matter, but it should be carefully aimed towards the world origin. As with the point light, you can add some visible objects around the light. Collect all components of the sun under a single hierarchy level.
2. Select the modeled sun object and click the SunMoon tool.
3. Adjust the date and the location.

Trim Curves

A trim curve is just a NURBS curve defined in the parameter space of a NURBS mesh. Trim curves are defined as sub objects for a NURBS mesh.

Any number of trim curves can be inserted into a mesh, in which case they are Boolean operated. The operator type can be selected from the mesh attributes.

Trim curves are derived from NURBS curves. In addition to attributes defined by NURBS curves, trim curves have the following attributes:



- **Cut** - if TRUE, the trim curve acts as true trim curve, trimming the parent. If FALSE, the curve doesn't trim the parent mesh. Such a curve is useful in 'welding'.
- **Invert** - If TRUE, the inside / outside property is inverted (controls whether the surface points inside or outside the curve are trimmed out).
- **WrapU, WrapV** - These attributes are meaningful only for trim curves of closed meshes. They tell the trim curve to close through the shortest route in the parameter space of the parent mesh. Thanks to this option, a trim curve can travel over the $u=1/v=1$ boundaries of a closed mesh.

Triset

Triset is a geometry, which consists of a number of triangular faces. Faces usually share the vertex data: connected triangles can refer to same 3D vectors. Each corner of a triset face has its own private normal vector. Normal vectors can be adjusted to achieve sharp or smooth looking shading over face boundaries.

Triset is much simpler object as the Subdivision object. It is also much more memory efficient. Its main function is to represent imported polygonal objects. If a triset geometry needs advanced editing, it can be converted to the SDS form, edited and then converted back to the efficient triset form.

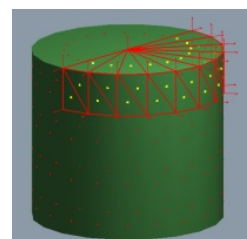
Triset property gadget has the following controls:

Faces: Displays the number of triangles in the selected triset object.

Points: Displays the number of vertices.

Selected: The index of the selected vertex in point editing mode.

Position: The position of the selected vertex in point editing mode.

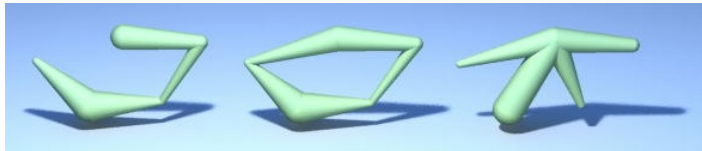


Phong Fix: Controls how rapidly illumination is faded towards the surface tangent. Normal interpolation of triset faces does not look smooth when ray traced illumination hits the surface at low angles, because the polygonal surface geometry starts casting shadows and blocking the illumination. By fading illumination at low angles, these shading errors can be made less visible.

Select: Edit mode selector. Trisets can be modified using point, edge and face handles. In face edit mode, selected faces display their surface normal handles, which can be edited using the mouse.

Tube

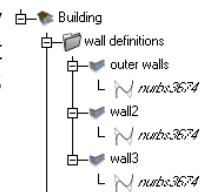
The tube object connects its sub objects (analytic spheres) with cylinders and cut cones, to form a continuous tube. It has only one object specific control: the **Type** of the tube, which can be open, closed or star like. The star alternative connects the first sphere to all other spheres.



The three tube types: Open, Closed and Star

Wall

The wall object defines a sequence of building walls. The object generates visible geometry only when placed into the 'wall definitions' sub folder of building object. The wall object needs one parameter object, a NURBS polyline. Currently smoothly bending NURBS curves define the wall by the control polygon.



Wall objects

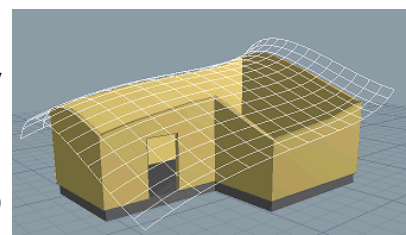
The sub wall object has the following options available in the Property window's Spec tab:

Wall Geometry options:

- **Thickness:** Thickness is wall specific. Outer walls are usually thicker than inner walls.
- **Side:** defines to which side of the parameter polyline the geometry is built.
- **Snap Start, Snap End:** If set, the respective end of an open wall segment is expanded to match the closest other wall. Room and building dimensions can be edited easily, because it is not necessary to adjust every wall dimension separately.
- **Virtual:** virtual walls divide space but do not generate visible wall geometry. They are considered in wall snapping and in room divisions. For example, it is possible to divide the floor of a room into two separate areas, which have different materials, by splitting the space with a virtual wall. Virtual walls also provide a way to cut staircase holes through ceilings and floors in buildings, which have several floors.

Roof Fitting options:

- **Extend to Roof:** when enabled, the wall height is adjusted automatically so that the wall joins with the roof seamlessly. Outer walls often need this option, whereas inner walls stop at the ceiling.
- **Wall-Roof Penetration:** A tolerance factor, which helps to close the gap between extended walls and the roof.



Walls extended to fit a curved roof

- **Fitting Interval:** The wall height is matched with the roof at every corner and where a roof line crosses the wall. If the roof has a complex or even a smoothly bending shape, a low fitting interval may be necessary. This option defines a maximal distance between two fitting positions.

Material Mapping Objects

Introduction

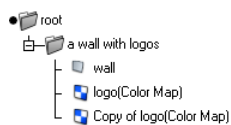
Material Mapping objects allow the user to assign material properties to geometric objects. They also allow post effects to be associated with objects. For example, one can associate a lens flare post effect with a set of 1D particles to get a flock of lens flares (perhaps acting under the influence of a gravity field).

The system follows the fundamental Realsoft 3D philosophy: a 'wooden sphere' consists of two sub objects, a sphere and a wood material mapping object.

Material Mapping Tool Set

Another idea behind the system is that different volumetric mapping methods, such as parallel mapping, spherical mapping, etc. are represented using a geometric object. For example, the material space for a spherical mapping is visualized using a sphere object.

This system has two remarkable features:



First, it allows any number of materials to be assigned per object.

Second, the tool set for editing geometric objects and material mapping objects is the same. One can move and rotate textures and other material effects as easily as one can move geometric objects.

A third advantage is that geometric objects visualize the material space in a very intuitive way. For example, the perspective projection is visualized using a pyramid object.

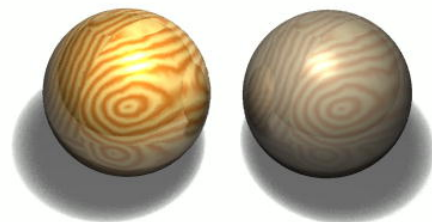
All material mapping objects define the following attributes:

Material - Name of the material to be mapped.

Effect - Name of the post particle effect to be mapped.

Translate, Scale, Rotate, Skew - Material coordinate space transformation that is applied after the mapping specific transformation is computed. For example, by changing the 'Scale' factors to 0.5, you can make a texture pattern 50 % smaller.

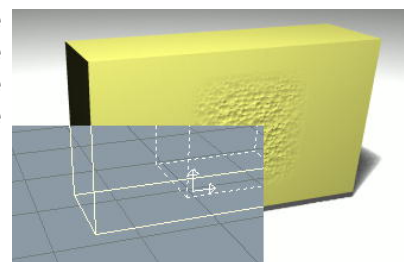
Scope - Controls how strongly the material affects the object.



Full wood and 'scope=0.5' wood

X, Y, Z Fading - Values greater than zero fade the material away at the both ends of the 0..1 unit interval of the respective mapping coordinate dimension. In other words, the material fades smoothly away at the mapping geometry edges. The value defines the width of fading area: value 1 will start fading from the middle point of the mapping. Value 0.1 will produce a 10 % fade area, 5 % at both ends.

Priority: Evaluation order control. By default, material maps are evaluated in the order they appear in the hierarchy, deep sub level materials last. The hierarchy defined evaluation order can be overruled with the mapping priority setting. The higher the priority, the earlier the mapping will be evaluated.



A bump map faded smoothly at parallel mapping edges, X and Y Fading = 0.5



Note

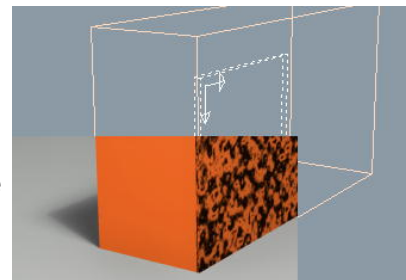
This means that the associated material will have a weaker effect! Later evaluated materials can overwrite the result. Some special materials, such as a global illumination shader, may work best if it is evaluated last. Because most material mappings have the default priority 0 and only some built in objects use evaluation priority -2 .. +2, a value -5 is usually sufficient to guarantee that the mapping is evaluated last. *Note 1:* The priority control is only rarely needed. Usually the hierarchy provides a much better way to control the order. However, in some cases, it would be too laborious to place the mapping to the end of all involved sub hierarchy levels. In such cases this feature can be a real time saver.



Note

The priority controls only material maps in the object hierarchy. SDS face materials don't use any mapping object and therefore they are always evaluated last.

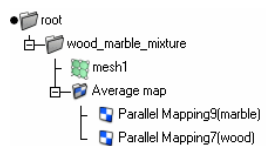
FiniteX, FiniteY, FiniteZ - If set, the material effect is limited inside the corresponding dimension of the mapping object (to 0..1 coordinate range)



A thin layer of stone mapped to a cube using a parallel map with the 'Finite Z' option set

Average Mapping

The average mapping object is a level object which uses any number of other material mapping sub objects. Material effects defined by the sub objects are evaluated and averaged and then assigned to the target objects. The Scope attribute can be used for weighted averaging, i.e. to control how strongly each sub mapping object contributes to the result.



Cube Mapping

The cube mapping object allows six different materials to be associated with an object - one for each side of the cube. This is particularly useful when setting up environment maps.

A cube mapping has the following object specific controls:

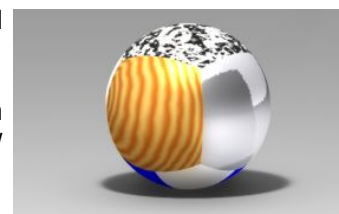
Volumetric Map: This cube mapping type uses 6 pyramid maps, one from each face of the cube into the center point of the cube.

Environment Map: The material is chosen using the reflected ray direction. This mapping can be used to simulate reflective surfaces inside a room interior.

Surface Normal Map: The material is chosen according to the surface normal direction.

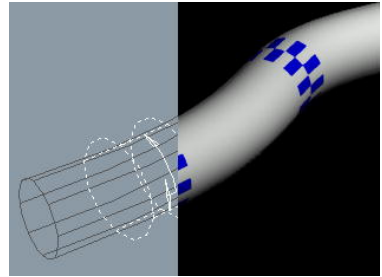
Front, Back, Top, Bottom, Left, Right Material: The six materials, from which one is chosen for the examined point using the selection rules (Volumetric/Environment/Normal) described above.

The geometric properties of a cube map are the same as those defined by the cube geometry object.



Cylinder Mapping

A cylinder mapping object can be used for wrapping a material around an object cylindrically. It uses a cylinder geometry object to define the material space.



Checkered stripes mapped to a tube using cylinder mapping

Default Mapping

A default map doesn't define any geometric properties. Instead, it uses the following five methods for defining the material space coordinates:

- Surface - The target surface defines the material space. This is a useful method when creating waving flags and skin textures, where the texture must stretch with the surface.



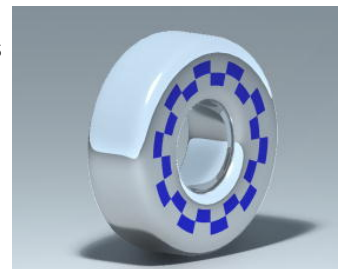
Note

Some surfaces do not define valid UV coordinates along the surface.

- Identity - This method uses the rendering space coordinates. Because the rendering space is tied to the camera, this alternative is suitable only for still images.
- Null - No material space computations at all. The material space remains as possibly defined by the previous material mapping objects. A quick mapping method for materials that do not contain any patterns, such as glass.
- Camera - Uses the current camera orientation to project material properties to target objects. A point under the top left corner of the image has coordinates (0,0,distance), the bottom right corner (1,1,distance).
- Custom - Moves a selected channel to the 'Map coords' channel. This is a suitable mapping type when you want to parametrize a material with a user channel that has been added as a 'vertex variable' to a SDS or NURBS object.

Disk Mapping

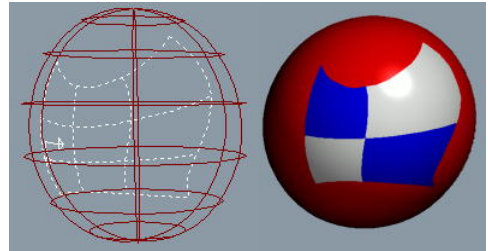
The disk mapping object uses an ellipse object to define the geometric mapping properties. It can be used, for example, to attach a 'Good Year' logo to the tires of a formula one car.



A checkered pattern disk mapped to a chrome wheel

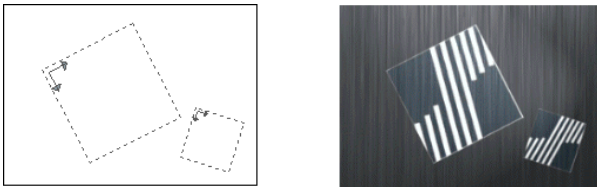
Mesh Mapping

A mesh mapping is a NURBS mesh that 'lives' on another NURBS mesh. In other words, the control points of the mesh map are UV values in the parameter space of the parent mesh. A mesh map can be created from the pull down menu 'Tools/Unified/Rectangular Object' and by setting the 'Class' option to 'MeshMap'.



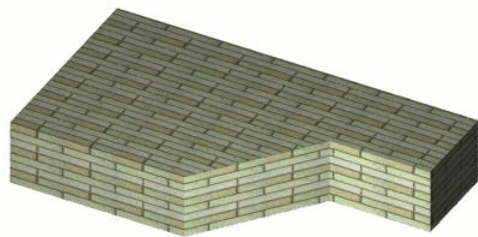
Parallel Mapping

A parallel mapping object uses a cube geometry object to define the material space.



Two logo textures attached to a rectangle using a parallel mapping. Ray traced image on the right.

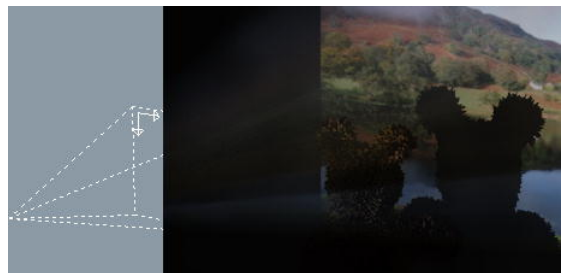
Parallel mapping has one special option: the **Roll** feature. The roll option rotates the Z axis of the mapping perpendicular to the surface normal, and other axes respectively. The option is most useful when texturing objects that consist of several planar surfaces. For example, one can map one brick pattern to one wall and then spread the pattern over all walls by using the roll option.



One single parallel mapping roll-maps all sides of a polyhedron

Pyramid Mapping

The Pyramid mapping object maps materials to geometric objects using a perspective projection. The attributes of a pyramid map object are the same as those defined by the pyramid geometry object. This mapping method is useful for creating slide projector effects.



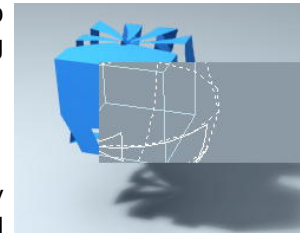
A slide projector created by pyramid mapping an illumination texture to a light source

Sphere Mapping

A sphere mapping object wraps a material spherically. It uses a ellipsoid geometry to define the material space. This mapping method is useful, for example, for defining environment maps.

A sphere map has one object specific option:

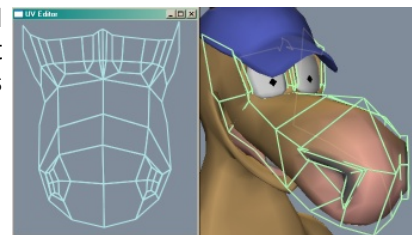
Environment Map: The mapping coordinates are computed from the reflected ray direction, not from the point position as usual. This mapping technique can be used for simulating reflective surfaces.



Spherical Clip Mapping

UV Set

UV Set is a powerful object for defining UV coordinates and material attributes for subsets of SDS objects. The power stems from the fact that the UV set object is derived from the SDS object itself. Many of the tools available for editing SDS objects are now also available for UV editing.



UV Set for texturing the upper half of a character's face

Creating UV Set objects

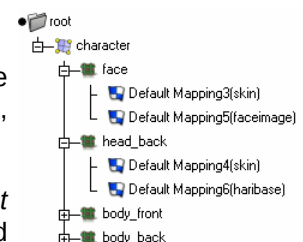
There are two ways of creating UV set objects:

1. Select an SDS object in the object edit mode and apply the *UV Set* tool with the *Cubic UV Set* option. This creates six UV-set objects representing front, back, up, down etc. directions.

2. Switch the SDS object into the edit mode, select faces and click the *UV Set* tool with the *Selected faces* option. This creates a single UV set of the selected faces. In this case, the UV coordinates for UV set vertices are fetched from the UV coordinates channel of the SDS object.

Of course, you can use edit/copy-paste and duplicate tools to create new UV sets from existing UV set objects.

UV set objects are created as child objects to the SDS object.



Surface division into UV areas and their texturing is managed efficiently using the select window

Any number of material mapping objects (parallel, spherical etc.) can be applied to an UV set objects to texture the actual SDS model. UV set also supports the direct texture definition in the property window's Col tab, as well as vertex channels, which now can be assigned memory efficiently only to sub areas of the SDS mesh.

Because the UV set object is SDS based, many SDS specific modeling tools can be used for editing. You can delete faces from the UV set, add new faces to it, join faces, unwrap and so on. All these tools are instantly available in the control bar as soon as you select an UV set.

UV editing tool set

In addition to the standard SDS editing tool set, there are also some UV set specific tools:

UV Set to Face Material

Converts selected UV sets to face materials of the parent SDS object. UV Set object's child mapping objects define the face material names. The UV values for the face materials are taken from the UV Sets. With this tool, you can convert the hierarchical UV set based texturing into a simpler form for exporting purposes.

Connect

This tool can be used for connecting two separate topological face groups into one.

From View

This tool takes the current view projection and uses it to set the UV coordinates of the selected faces of the UV set. Select some faces, navigate on the view window and click the tool. Simple and quick.

Light Sources

Realsoft 3D supports a rich set of light sources. The light source attributes described below include quite extensive controls. Even more lighting power can be achieved by using materials that define custom lighting effects using a Light properties shader.

The controls that are available for most light source types are listed below.

Intensity: Controls the brightness of the light source. The intensity value of a light source is combined with its RGB color value. For example:

intensity=0.5, color=(0.8, 0.7, 0.6)->light color=(0.4, 0.35, 0.3).

Volumetric: If set, the light source is considered in volumetric lighting. Fog materials that have diffuse color become illuminated by this light source.

Diffuse Only: If set, the light source creates only the default diffuse shading i.e. the illumination is proportional to the angle at which light hits the surface. Specular lighting and other VSL defined shading effects will be ignored.

Falloff Frame

These controls define how the illumination fades as the distance from the light source increases.

- *None* - Illumination does not depend on the distance.
- *Local* - Illumination fades to zero within the falloff radius.
- *Distance* - Illumination is inversely proportional to the distance. Light intensity decreases to 50% when the distance is doubled.
- *Distance squared* - A physically correct falloff for point light sources. Intensity decreases rapidly by distance.

Radius: The falloff distance. Local falloff drops to zero at this distance, 'Distance' and 'Distance squared' falloffs decrease to 50 % at this distance.

Shadow Casting Frame

Shadow casting properties are available for all light source types. Point and spot lights support shadows mapping.

No shadows: The light source does not cast any shadows. This kind of light source renders quickly.

Ray traced shadows: Shadows of the light source are computed using ray tracing. Such shadows are realistic and obey material properties (glass object do not cast black shadows etc.).

Mapped shadows: Shadows are first rendered to a bitmap which is then filtered in a suitable way. This kind of shadows have smooth edges and often render quickly. However, mapped shadows are more inaccurate than ray traced shadows and do not support shadow translucency.

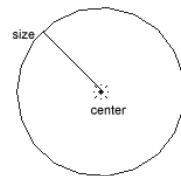
Resolution: The accuracy of the shadow map. Both x and y dimensions use the same pixel resolution. A low resolution, for example 200 pixels, will make shadow computations quick but inaccurate. A higher resolution, for example 1000 pixels, is accurate, but shadow map rendering takes much longer time.

Point Light

A point light source is defined by a center and a radius. A point light illuminates light to all directions. The radius attribute can be used for controlling the fall off.

It is also possible to set a 'size' attribute, in which case the point light becomes an area light generating soft shadows in a physically correct way.

Point light also supports an 'Ambient' option. If set, the light source no longer emits light rays radially but randomly to all directions. The position and size attributes are then used for controlling the strength of illumination. This allows the user to create local ambient lighting effects.

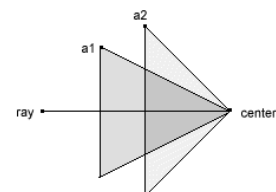


Point light specific options:

- *Point/Ambient light* - The Ambient option turns the point light into an ambient light.
- *Position* - The position of the light
- *Size* - The radius of the light source. Size is used when computing mapped shadows. It is also used for ray traced smooth shadows i.e. when 'Quality' is greater than 0. The higher the size, the smoother the shadows become.
- *Quality* - The sampling quality for ray traced smooth shadows. A high quality value increases rendering time but produces smooth shadows, whereas low values render quickly but may introduce some dithering artifacts at shadows edges. The bigger the light source size, the higher quality value is needed for a proper quality.

Spot Light

A spot light is defined by two light cones. The intensity of the light drops gradually to zero between the inner and outer angles. A spot light source is often quicker to render than a point light or a distant light because the illuminated area is more compact.



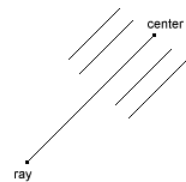
Spot light specific options:

- *Position* - The position of the spotlight
- *Aimpoint* - The point at which the spot is aimed

- *Angle1, Angle2*: - The angles that define the spot beam. The greater one of the two angles defines where spot illumination ends. Illumination fades from the full intensity to zero between the two angles. If the angles are equal, the spot light beam has a sharp edge.
- *Material* - This control includes a list of the fog like materials of the current material library. Selecting a material will render a fog cone that matches the spot beam. This provides an easy way to make the spot beam visible.
- *Size* - The radius of the light source. Size is used when computing mapped shadows. It is also used for ray traced smooth shadows i.e. when 'Quality' is greater than 0. The higher the size, the smoother the shadows become.
- *Quality* - The sampling quality for ray traced smooth shadows. A high quality value increases rendering time but creates smooth shadows, whereas lower values render quickly but may introduce some dithering artifacts at shadows edges. The bigger the light source size, the higher quality value is needed for a proper quality.

Distant Light

A distant light is defined by two points: center and ray. These two points define the direction in which the light source illuminates. All light rays are parallel to each other. A distant light can be used for simulating distant light sources, such as the sun. At rendering time, the program moves a distant light source internally infinitely far away from the target. Therefore, no light fall off settings are available.



Special Light

A special light source is a level object which doesn't define any geometric data. However, it allows the user to use any geometric object as a source of light; all sub objects radiate light. If the sub object is a single point, a point light is achieved. By using a NURBS curve as a sub object, one can create one dimensional area lights. Using a rectangle creates a two dimensional area light. The colors of sub objects of a special light sources are taken into account, including texture mapped colors in a 'Light properties' shader.

Special light specific options:

- *Quality* - The illumination sampling quality. Defines how densely the sub geometry surfaces are evaluated for estimating the total illumination. A high quality value increases rendering time but produces smooth shadows, whereas low values render quickly but with dithered shadows. The bigger the sub object are, the higher quality value is needed for a proper quality.

Ambient Light

An ambient light is actually a point light with the 'Ambient' option set. The point geometry can be used for falloff purposes; for example, you can add a local ambient light source in front of a TV screen to simulate the local illumination it creates, or a local negative ambient light source in a dark corner of a room scene.

Usually ambient light sources work best if the Diffuse option is set and shadows are turned off. Both options are allowed for special effects purposes.

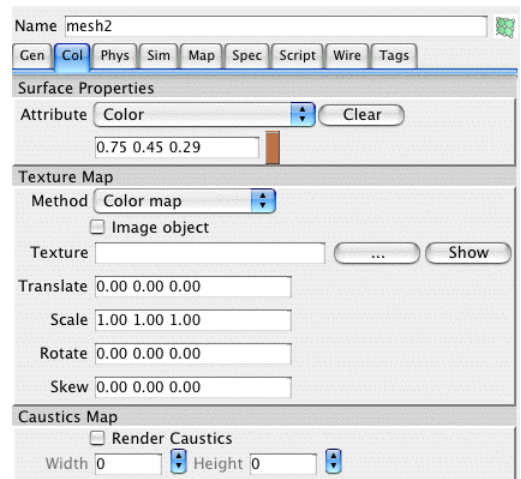
Common Properties of Geometric Objects

Color Tab

The Col tab contains color and other surface property related settings. These controls provide a simple and easy way to modify surface appearance (shading) of objects. When more control is needed, VSL materials provide a more powerful way to define shading properties.

Surface attributes defined in the Col tab are evaluated before the materials assigned to the object. This has two consequences:

- Surface attributes are 'weaker' than material assigned attributes. A color defined by a VSL material overrules a color defined here.
- Materials can utilize Col tab attributes. For example, a glass material can derive the transparency from the usual object color attribute. Then you can easily create different colored glass objects by using the same material and changing the object color.



The Surface Properties Frame

Attribute: This gadget contains a list of ordinary surface attributes. Color is the most commonly needed attribute, but you can also define Illumination (=self illuminating color), Reflection, Transparency, Fade, Alpha etc. properties here. For more information about channels, check the Channels documentation.

Clear: Removes the value for the attribute defined by the previous field from the object. The object will then use a value possibly defined by a parent level, or if the parent value is not set, the default value of the attribute.

Attribute Value: This field controls the value of the selected attribute. A floating point, vector or color gadget is displayed depending on the attribute type.

The Texture Map Frame

Method: Selects the type of the texture mapping. A color map defines the usual diffuse color. 'Environment map' makes the object mirror like (black diffuse color, full reflectivity) and uses the defined texture file as a spherical reflection map for the surrounding world.

Texture: The name of the image file for texturing.

Translate, Scale, Rotate, Skew: These four gadgets define a texture transformation matrix. For example, the texture can be tiled and repeated several times by decreasing the two first components of the Scale vector.

The Caustics Map Frame

Render Caustics: Enables caustics rendering for the object i.e. makes the object receive caustics illumination. Note: the top level caustics switch in the render settings must be also enabled.

Width, Height: The accuracy of the caustics effect for this object. Caustics illumination is stored to a object specific illumination map, and these values define the resolution of the map. The higher the resolution, the finer details caustics illumination contains. However, high resolution usually requires that the caustics Sampling value is high enough - otherwise the illumination details may become too weak or noisy. A good thumb rule is that Width and Height are about the sampling value multiplied by ten. For example, the default sampling rate 20 is enough for a 200*200 caustics map.

General Properties

The General Properties tab includes some frequently needed properties of geometric objects.

Name: The name of the selected object(s).

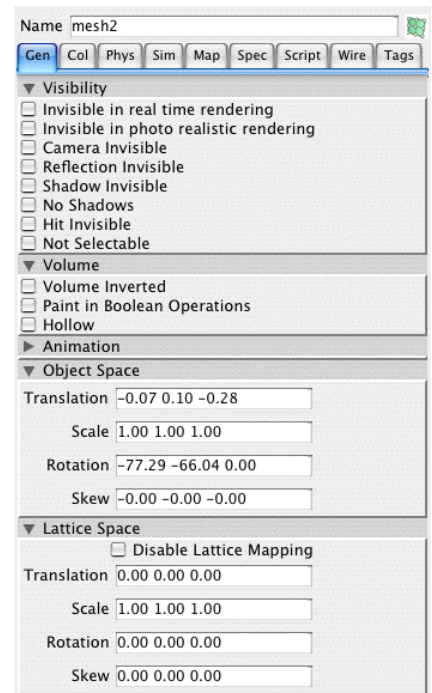
Class Icon: At the right side of the name gadget, a small icon shows the class of the selected object. The associated tooltip tells the name of the object class (rectangle, cube etc.).

The Visibility Frame

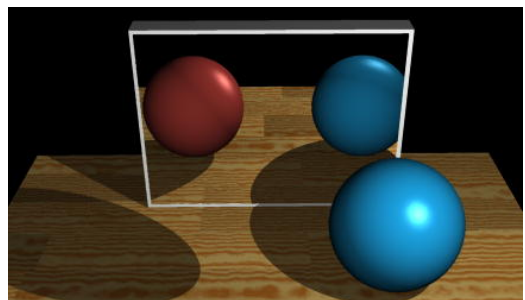
Miscellaneous visibility options.

Invisible in real time rendering: Hides the object (and its sub objects) from real time view rendering (both GDI and OpenGL).

Invisible in photo realistic rendering: Removes the object from ray tracing and other 'final' rendering systems.

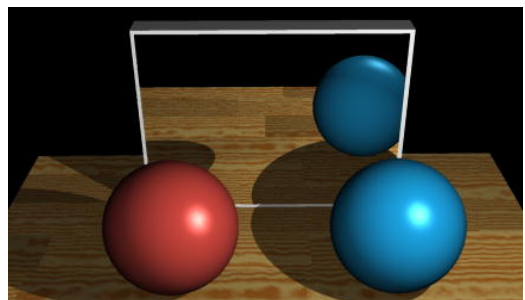


Camera Invisible: Makes object invisible for the ray tracing camera. Invisibility covers only the direct view from the camera: the object remains visible via reflections, casts shadows etc.



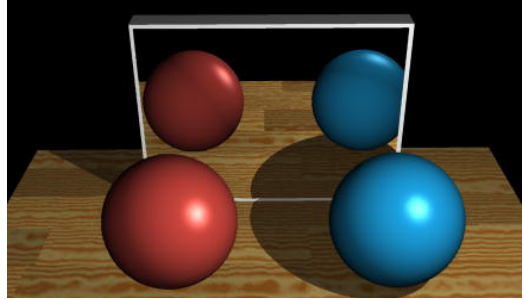
A camera invisible red sphere in front of a mirror

Reflection Invisible: Object will not show in reflections or through transparent objects.



The red sphere is now reflection invisible

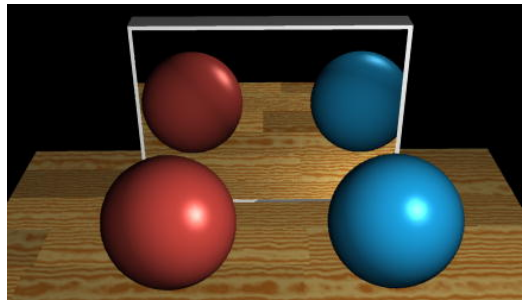
Shadow Invisible: If set, the object will not cast any shadows. This can be used as a rendering optimization trick. For example, room walls can be made shadow invisible if the camera and all light sources are inside the room.



The red sphere made shadow invisible

No Shadows: If set, the object will not receive any shadows cast by other objects or the object itself. Light sources do illuminate the object; only the shadows are eliminated.

Hit Invisible: If set, the object will be ignored in collision detection, 3D painting and in the surface projection of various tools such as the particle brush tool and the paste tool. For example, you can make a water surface hit invisible before brushing some rock particles into the bottom of a lake.



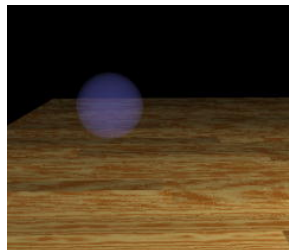
No Shadows set for the floor

Not Selectable: If set, the object (and possible sub objects) cannot be selected by clicking or dragging the object wireframe on a view window. The object can still be selected from the select window. The option is useful when editing complex scenes with lots of wireframe.

The Volume Frame

Volumetric object properties for Boolean operations and volumetric rendering effects.

Volume Inverted: The object fills the exterior of its natural volume with matter. For example, a tiny sphere becomes an infinite space with a small bubble in it.

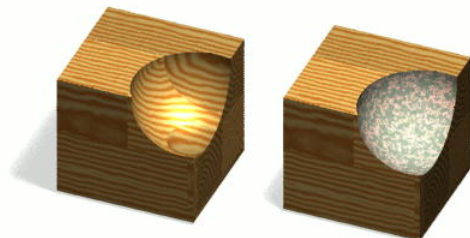


Fog bubble



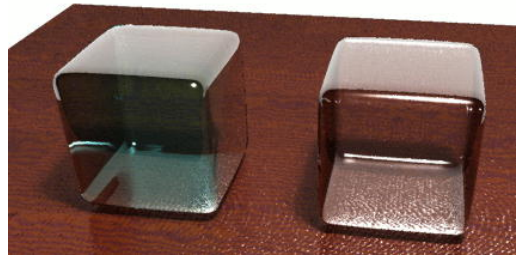
Inverted fog bubble

Paint in Boolean Operations: If set, the object keeps its original surface attributes when cutting another object. By default, the first sub object of a Boolean AND operation level defines the properties of the result.



Wooden cubes cut with a marble sphere. The marble cutter at the right has the 'Paint' option enabled.

Hollow: Makes an object hollow i.e. an infinitely thin surface without a volume. This shows up when Boolean cutting the object or if the object is made of transparent, refractive matter.



Two glass cubes made of the same material. The one on the right is hollow

The Animation Frame

Object behavior in animations.

Rotate in HPB: By default, object rotations are animated using so called Quaternions, which produce rather intuitive smooth rotations. This option selects a more machine like rotation interpolation in the Heading-Pitching-Banking angle space.

Animate only when selected: Objects can be animated by modifying their animation recording on. Sometimes objects change (because of a lattice map connection etc.) even when they are not selected. This option helps preventing unintended animation event recording for unselected objects.

Animation Protected: If set, the object will not record any key frames when modified in animation recording mode. This helps to avoid unintended key frames.

Key frame animation only: This option is similar to the previous one, with the exception that already animated attributes do record new key frames. Unanimated attributes remain unanimated.

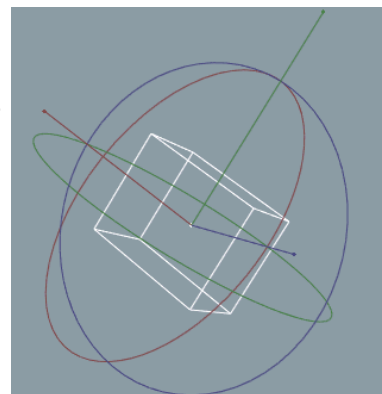
Hide Choreographs: Hides all choreographs associated with this object and its sub objects. This option is useful when managing scenes consisting of thousands of animated objects and you don't want to see them all in the Choreography Window.

The Object Space Frame

Object space transformation components. You can see the object space position and orientation on view windows through the coordinate handle which appears when you select the object. The X axis is drawn in red color, the Y axis in green color and the Z axis in blue color. You can remember the color order easily by checking the view window orientation guides - each axis is labeled there.

Translation: The position of the object space origin (pivot point).

Scale: The total scaling of the object (measured along object space axes).



Rotation: The total rotation of the object. The three values are the heading, pitching and banking angles which rotate the world space XYZ axis system to the current object space axis system as follows:

- Heading angle first rotates the world X and Z axes around the world Y axis.
- Pitching angle then rotates the obtained Y and Z axes around the rotated X axis.

- Banking angle finally rotates the obtained X and Y axes around the rotated Z axis.

Skew: The shearing angles of the object space. If all angles are 0, the object space is orthogonal.

The Lattice Space Frame

The transformation components of a lattice mapped object. A lattice mapped object has an additional transformation matrix which specifies the object orientation in the lattice space. For example, making a curve mapped object to follow the path is simply a matter of animating the lattice space translate x component to increase from zero to one.

Disable Lattice Mapping: If set, modifying a lattice object to which the object is mapped no longer affects the object.

Translation: Object position in a lattice space. For example, if you place the pivot point of an object to the beginning of a path curve and then map it, the lattice translation will be (0,0,0). Lattice mapping an object to the middle of a mesh will set the translation to (0.5, 0.5, 0).

Scale: Object scale in lattice space.

Rotation: Object rotation in lattice space.

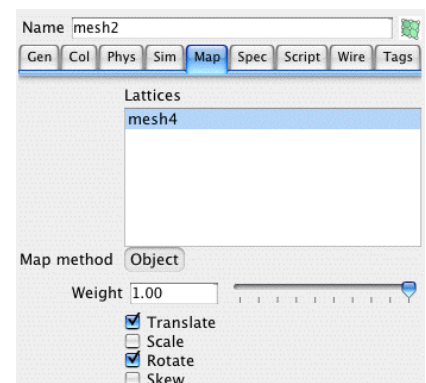
Skew: Object shear in lattice space.

Mapping Tab

The Map tab shows the lattice mapping summary for the object.

Lattices: The list of objects to which the edited object is lattice mapped to. You can click a list item to view the properties of the corresponding lattice binding at the bottom of the property window. The list also includes a popup menu 'Remove' for deleting undesired lattice connections one by one.

Map method: The type of the lattice mapping, Pointwise or Object. The pointwise mapping means that each geometry point is treated individually in lattice mapping. Object mapping controls the object via its object space that is attached to the lattice.



Lattice mapping properties of an object following a directed path

Weight: Defines how strongly the lattice mapping affects the target. Weight one means that the control is total i.e. the mapped object fully obeys the lattice. A weight zero disconnects the object fully from the lattice. When using several lattices, a weighted average is computed. Note that in pointwise mapping, also the weight is pointwise. To control pointwise weights, put the mapped object to the point edit state, select some points and enter the desired weight.

Translate, Scale, Rotate, Skew: The transformation components of an object space type lattice mapping. If only Translation is set, the lattice controls only the object space position of the mapped object. If also Rotate is set, the lattice controls the orientation of the target. The Scale component has importance when the lattice is two or three dimensional. For example, if an object follows a mesh and the scale option is enabled, the object will shrink together with the mesh. Finally, the Skew component allows a non orthogonal lattice control making the object to shear together with the lattice.

Physics Tab

The 'Phys' tab includes controls for physical object properties which are used in simulations.

General Properties

Mass: The mass of an object (in kilograms). The default mass is 1 kg. A level object's default mass is the sum of the mass of sub objects. Any other mass can be defined here.

Clear Mass: Clears a custom mass definition so that the default mass computation will be used.

Electric Charge: The amount of electric charge (coulomb). Can be positive or negative.

Surface Friction: A surface friction coefficient 0..1.

Rebound Energy: Describes how well the object keeps its kinetic energy in collisions. The value range is 0 .. 1. If the value is 1, the object keeps its kinetic energy and just changes the direction of its motion in a collision.

Elasticity: Objects ability to maintain its shape in a deforming situation such as a collision. If elasticity is zero, there is no force that restores the shape after a deformation.

Rigidity: Describes how rigid the object shape is. The lower the value, the more freely object matter can react to forces and the less internal friction resisting the shape deformations exists. If the rigidity is zero, object geometry points act independently in simulations (elasticity defines a string force that maintains the shape).

Inertia: Describes how the mass is distributed in object volume. If the mass is concentrated in the middle of the object, the inertia is small and the object has a tendency to spin easily in collisions. If the mass is concentrated at outermost edges (say a hollow lead sphere), the object has a large resisting momentum and does not spin easily. The program computes a default estimate from the object mass and bounding box dimensions. Be careful when changing the default momentum; entering a value that is physically impossible will invalidate the collision impact calculations.

Clear Inertia: Removes a non default inertia definition. The value estimated by the program will be used.

Center of Gravity: The program estimates the center of gravity as an average of the surface geometry, but you can change it here if necessary.

Clear COG: Removes a non default COG definition. The value estimated by the program will be used.

Velocity: Defines the direction and speed of object's motion.

Spin: Defines how the object rotates around its object space axes. The unit is the selected angle unit per second. For example, (0, 180, 360) degrees per second.

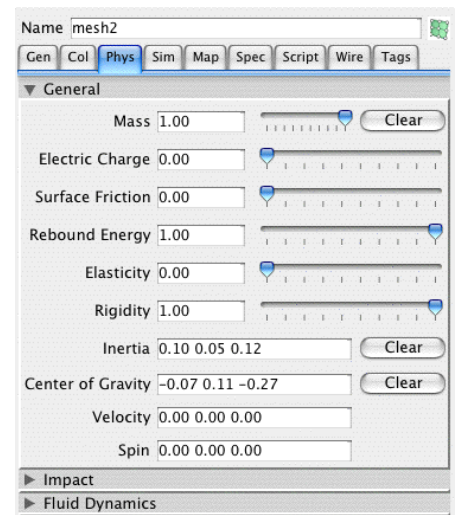
Impact Properties

The impact system maintains an object attribute 'Impact Level', which can be used for key driven animations launched by collisions. For example, you can key frame a 'collision behavior' and bind it to the Impact Level attribute.

Impact Duration: The duration of impact wave (measured in the selected time units - frames, seconds etc.). The impact wave runs linearly from zero to Amplitude or vice versa during the duration time.

Impact Curve: Selects if Impact Level increases (0 to amplitude) or decreases (amplitude to 0) within the impact wave duration time.

Amplitude: The maximal value of the Impact Level attribute during the impact wave processing.



Impact Level: The current value of Impact Level. You can use this control to test a choreography driven by the Impact Level attribute.

Fluid Dynamics Properties

Density: Describes how dense a matter where simulation happens is. A typical value range is 1..10. The denser the matter, the more it resists motions (or creates them in case the matter has a moving flow such as wind). This property is attached to objects that define a fluid such as the fan object.

Fluid Velocity: The speed of fluid flow as meters per second (the flow direction is defined by the fluid creator geometry, for example a fan axis). Also this property is attached to objects that define a fluid flow such as the fan object.

Fluid Pressure: The pressure of fluid. This value affects e.g. forces generated by a non symmetric fluid flow.

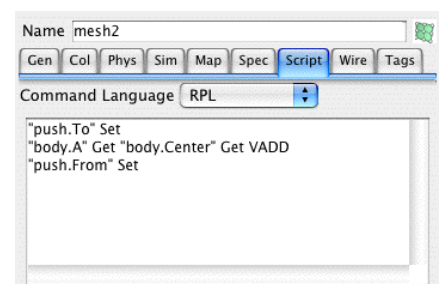
Fluid Friction: Object's friction coefficients in object space directions. Many objects are strongly non symmetric in this respect: for example, an airplane is designed to have a small friction in its natural peak-tail direction, but it has quite a strong friction against the wing planes. This property controls objects moving in fluid, not the fluid creator.

Fluid Symmetry: Fluid flow symmetry with respect to object space axes. The value zero means symmetric flow. Non symmetric flows generate a pressure difference at the opposite sides and hence a force along the axis in question (for example: an airplane wing).

Scripting Tab

It is possible to control object behavior in animations using scripts. The Script tab contains an editor for this purpose. Scripts entered through the Script tab are executed last in the construction stack so they can be used for controlling other animation effects easily.

The **Command Language** gadget shows all installed script languages. With it, you can choose which language to use for the object.



You can enter the actual script to the text editing area below the language selector. For example, the following JavaScript program:

```
Self.SetCenter(new r3Vect(0, 0, 0));
```

sets the center point to zero. The variable 'Self' refers to the object the script is associated with and SetCenter() is a method defined by the class in question.

Consult the Realsoft 3D JavaScript reference manual for more information about Realsoft 3D classes and the methods they define.

The following script controls the size of the sphere using the color attribute.

```
color = Self.GetColor();
Self.SetTotalTranslate(color);
```

To make sure the object don't go below 'y=0' level:

```
t = Self.GetTotalTranslate();
if(t.y < 0) {
    t.y = 0;
    Self.SetTotalTranslate(t);
}
```

```
}
```

'Time' variable holds the current time allowing you to create animations easily. For example, to control the object's color in function of time:

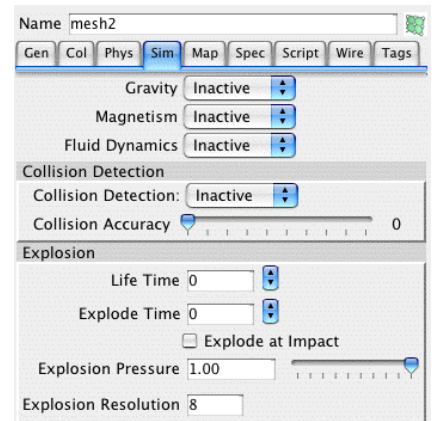
```
Self.SetColor(new r3Vect(Time, Time, Time));
```

Simulation Tab

The 'Sim' tab includes switches for turning various simulation calculations components on and off.

Gravity: Enables Gravity force for the selected objects. There are 4 options available:

- *Inactive* - The object is not considered in gravity computations.
- *Both* - The object creates a gravity field and is pulled by gravity forces.
- *Cause* - The object generates a gravity field. It does not react to gravity forces.
- *Affected* - The object does not create gravity, but is pulled by gravity forces.



Magnetism: Enables Electro Magnetic simulation. Magnetism is different from gravity in that it is a signed quantity: similar charges repel each other, opposite charges attract each other. The four magnetism alternatives - Inactive, Both, Cause, Affected - work the same way as in the Gravity setting above.

Fluid Dynamics: Enables fluid simulation. See 'Gravity' above for the option types. Typically only fluid specialized objects such as a fan have the 'Cause' option set i.e. they generate a fluid environment. Other simulated objects use 'Fluid Dynamics = Affected' option.

Collision Properties

Collision Detection: Enables Collision detection as follows:

- *Inactive* - The object does not collide or make other objects collide.
- *Both* - The object creates collisions and collides itself.
- *Cause* - The object creates a collision impact when hitting other objects. However, it does not react itself to the collision.
- *Affected* - The object reacts to the collision impact. However, it does not change the motion of other objects in a collision.

Collision Accuracy: Defines how accurately the collision point is examined. The higher the value, the sooner collision (= object volume penetrating another object volume) is detected. High values naturally increase computation time. If the object shape is complex (includes sharp extrusions etc.), a high collision accuracy may be necessary to prevent object volume overlapping.

Explosion Properties

Life Time: The life time (in the current time unit frame, second, ..) of the object. The object is removed from the scene after the life time. Zero value has a special interpretation: it does not kill the object immediately but disables the feature.

Explode Time: The time after which the object explodes measured using the current time unit. Zero value has a special interpretation: it does not explode the object immediately but disables the feature.

Explode at Impact: If set, the object explodes at a collision impact (collision must be enabled).

Explosion Pressure: Defines how quickly the object parts move apart because of the explosion.

Explosion Resolution: The amount of parts (density of object subdivision) generated by the explosion.

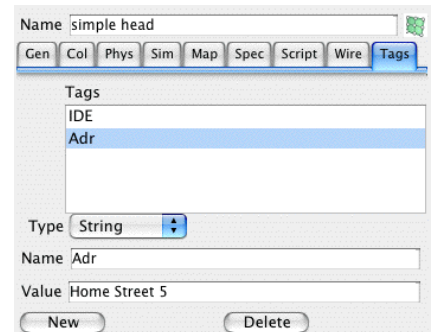
Tags Tab

You can attach additional information to objects using tags. Tag information is visible to the animation system, so you can key frame your custom tags or use them to parametrize animations.

The **Tags** gadget shows all user defined tags of the object. Clicking a tag item activates it showing the tag properties at the bottom of the window.

The **Type** gadget sets the data type of the selected tag. All standard data types such as integer, floating point, color, vector, string etc. are available.

The **Name** field sets the tag identifier. A tag identifier is a three digit character sequence. New tags use a dummy identifier 'XXX', which can be renamed as appropriate.



The **Value** field controls the tag value. The gadget adapts automatically to the selected data type.

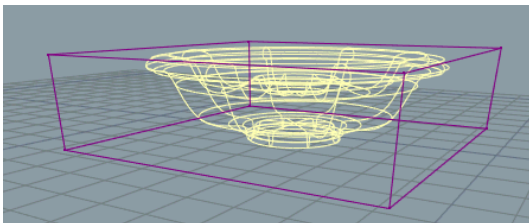
The **New** button adds a new tag item to the tag list.

The **Delete** button removes the selected tag item from the tag list.

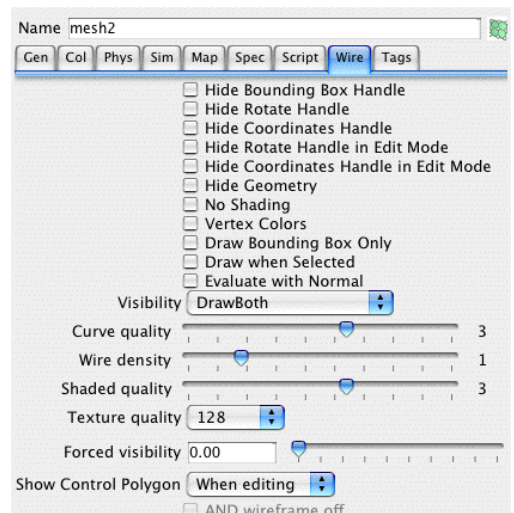
Wire Tab

The Wire tab contains object settings that are related to real time drawing, both in GDI wireframe and using shaded/unshaded OpenGL. Most of these settings are common to all objects. Some object types insert additional controls to the bottom of the window; check the object specific documentation for more information.

Hide Bounding Box: Hides the bounding box edit handle of this object.



Bounding box handle drawn with a purple color



Note

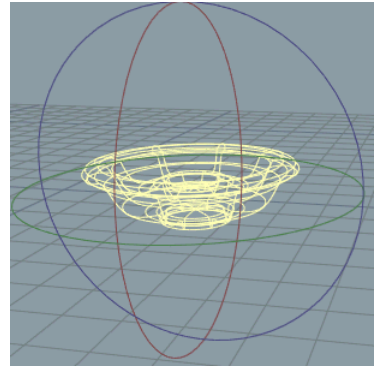
Bounding box handles do not show unless their drawing is enabled from the view popup menu.

Hide Rotate Handle: Hides the object space rotate handle of this object.



Note

Rotate handles do not show unless their drawing is enabled from the view popup menu.



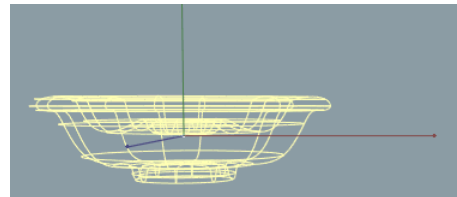
Rotate handle shown, other handles hidden

Hide Coordinates Handle: Hides the object space scale and translate handle of this object.



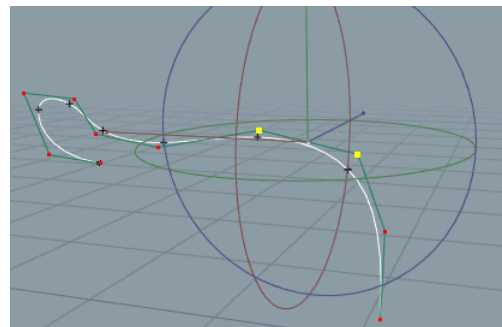
Note

This handle type does not show unless the rotate handle drawing is enabled from the view popup menu.



The Coordinates Handle

Hide Rotate Handle in Edit Mode: Hides the rotation handle of this object in point editing mode. Some objects, such as the Subdivision Surface, display so many other handles that hiding the default handle may be necessary to avoid excessive handle wireframe. With the rotation handle, as well as the Coordinates Handle described below, you can edit the selected points in object's space. The handles are positioned in the average point of the selected points. Rotate handle drawing must be enabled from the view popup menu, otherwise the handles won't be drawn.



Point editing a NURBS curve. The rotate and coordinate handles are enabled.

Hide Coordinates Handle in Edit Mode: Hides the scale/translate handle of this object in point editing mode. You can edit the selected points in the object space with this handle. Rotate handle drawing must be enabled from the view popup menu, otherwise the view window will not draw the handles.

Hide Geometry: Makes the actual object geometry (surface, curve, etc.) invisible in real time drawing. Handles and possible sub objects will be drawn normally.

No Shading: Forces wireframe drawing also in shaded OpenGL drawing.

Vertex Colors: If set, colors from materials are evaluated per mesh vertex. This overrides Texture Quality setting - no texture bitmaps are used. This is sometimes the most suitable and quickest way to roughly show the surface colors.

Draw Bounding Box Only: Hides the actual surface and shows only the bounding box of the object. Sometimes it is enough to see only the objects under editing using accurate geometric representation. Other objects can be drawn using simple reference boxes. Using a bounding box instead of accurate wireframe or shaded surface naturally speeds up model drawing and reduces visual complexity of a detailed scene, thus making editing the scene easier.

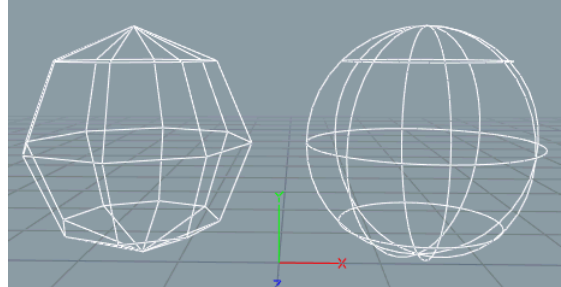
Draw when Selected: If set, object wireframe becomes visible only when the object is selected directly (not as a part of selected parent level). This is useful for example for material mapping objects, whose wireframe does not clutter the scene, but which become visible when it is time to adjust the size or proportions of the mapping.

Evaluate with Normal: When set, shader conversion to OpenGL textures computes surface normal. This gives more accurate result when shaders (or material maps such as the roll mapping) depend on the normal.

Show Name: If set, the object name is displayed on view windows.

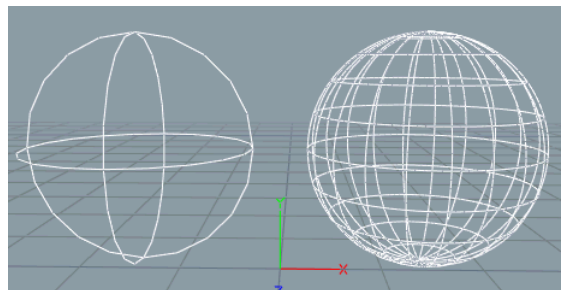
Visibility: Selective visibility in shaded/unshaded drawing. For example, the standard shaded mesh along the object's surface can be used for OpenGL drawing and another simplified 'symbolic' wireframe for GDI drawing.

Curve quality: The density of wireframe curve subdivision. Higher values display curves more accurately, but increased density makes drawing slower.



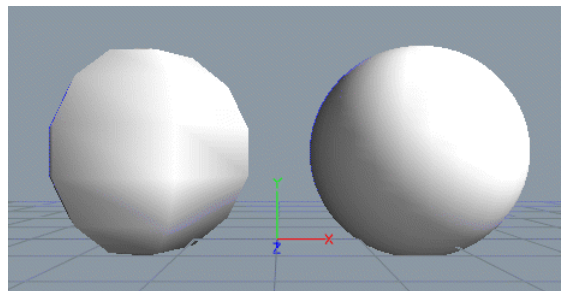
Two identical spheres. The sphere on the right is drawn using high curve quality.

Wire density: The density of wireframe mesh i.e. the density of wireframe curves drawn on surfaces. This setting has no effect on 1D curves.



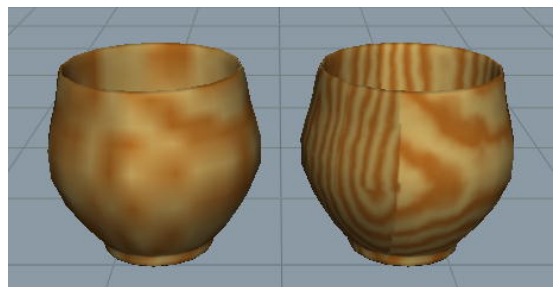
Two identical spheres. The sphere on the right is drawn using high wire density.

Shaded quality: The surface drawing quality in shaded OpenGL. Increasing the value makes surface tessellation to polygons denser, so that the surface appears smoother.



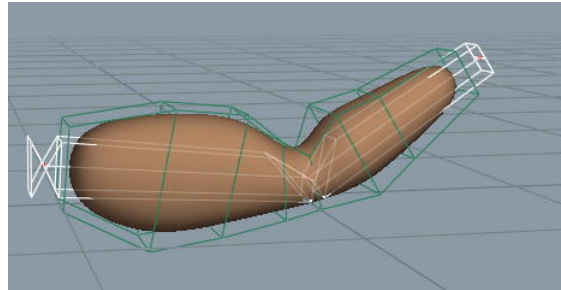
Two identical spheres. The sphere on the right is drawn using high shaded quality.

Texture quality: The quality of material evaluation for shaded OpenGL. The material system of Realsoft 3D supports blending of multiple (mathematically defined or image mapped) materials. The OpenGL drawing system can evaluate the result as a single texture map that colors the surface. The values of this control describe the pixel resolution of the result texture. Evaluation of complex 'final rendering' oriented shaders can be time consuming, so be careful when applying high quality values. Note: 'direct' textures assigned in the 'Col' tab can be drawn very quickly and therefore such textures are drawn in the original resolution (regardless of the texture quality setting in the wire tab).



Two cups with a procedural wood material. The left one has a texture quality 32, the right one 256.

Forced visibility: When using shaded OpenGL drawing, it is often important to see certain items even when they are behind surfaces. For example, object editing handles are always somewhat visible through obscuring surfaces. Sometimes complete objects must be seen as well: a skeleton is placed inside a character object. Its visibility is important when editing the character pose. A suitable forced visibility value makes the skeleton always visible.



A skeleton forced to show through the flesh

Constructors

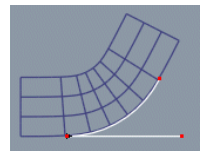
Bend Object

Bend deformer is a constructor based deformer. It can be used for generating circular bend effects to free form objects.

The tool asks the user to define an axis with specified length. This axis can then be bent and all the target points will be deformed accordingly.

To bend a nurbs mesh:

1. Create a nurbs mesh consisting of 10 x 6 control points
2. Select the mesh and activate the bend tool. Then define bending axis along one of the edges of the nurbs mesh.
3. Switch the created bend object to point edit mode and drag the end point handle to bend the axis. The nurbs mesh is bent with the axis.



A bend deformer applied to a nurbs mesh

Bending radius can be specified numerically through the property window.

Displacer

Point Displacer deforms geometric objects using material mapping objects.

The displacement effect is defined by one or several material mapping objects, which are sub objects for the displacer.



A displaced mesh consists of a mesh and a displacer object

When using multiple materials for displacement, the selected materials are responsible for combining the computed channel values. A working combination could be:

<code>material1</code>	<code>material2</code>
<code>surface properties</code>	<code>surface properties</code>
<code>color=*noise(map coords)</code>	<code>color=*curve(map coords)</code>

The shaders below will not combine well (only the latter material mapping counts):

<code>material1</code>	<code>material2</code>
<code>surface properties</code>	<code>surface properties</code>

color=noise(map coords)

color=curve(map coords)

Point Displacer has the following controls:

Channel: Selects the material channel, which defines the displacement.

Space: Orientation for moving the target points.

- *Absolute* - The world space defines displacement direction. If the 1D option is set (or the selected channel is one dimensional), displacement happens along the Z direction of the world space. Otherwise the first sub channel component defines X displacement, second sub channel Y displacement etc.
- *Object* - The object space of the point displacer itself (not of the target mesh!) defines the orientation for displacement. For example, if 1D option is set, the target mesh is displaced along the blue Z axis of the displacer space.
- *UVW Space* - The surface orientation of the original non-displaced mesh defines the displacement direction. In 1D case, the target surface is displaced perpendicularly i.e. along its normal.

1D: - If set, the target mesh is displaced only in Z axis/Surface normal direction. If the selected channel is 3 dimensional, the sub channel average defines applied 1D displacement values. 1D displacement is usually the most intuitive and easiest deformation available via this constructor.

Carriage Object

Carriage object is defined by a coordinate system. The coordinate system defines a linear transformation that is applied to the attached target objects.

Carriage object defines the following attributes:

Origin: The origin of the carriage coordinate system.

X, Y, Z: Three axes of the carriage coordinate system.

Drag Object

To use the tool, multi select an animated skeleton, a "floor" object and enter two points through the view window to define gravity. The drag object will then study the movements of the skeleton and translate the skeleton accordingly. Bodies you bind to the skeleton will move with the skeleton.

Or drop the skeleton and the body into a single level and apply the drag tool to the level. In this case also the body points will be taken into account when the drag object computes translation.

Drag object uses collision detection system to determine how much the target object should be moved based on the pointwise translations defined by the target. For example, animate the legs of a robot and drop the robot on a floor. Drag object will make the robot walk on the floor according to the pointwise velocities of the legs.

Drag object affects only the translation of the target objects. It doesn't generate spin to the target objects.

Drag object defines the following attributes:

Rebound: Specifies rebound energy for the collision detection. If the rebound energy is zero, object will not rebound from the surface.

Friction: Surface friction. Setting friction to zero results totally slipping feet effect. Setting friction to max (1) will eliminate slipping feet effect completely.

Inverse Kinematics Object

This is a special purpose geometric sphere that applies inverse kinematics method to the joint it is associated with.

The idea is that one can define independent, reusable motion libraries and associate them with different type of skeletons.

To use this object:

- Select a skeleton and create I.K object over a joint. The created object is automatically bound to the joint inside it. Note that binding is based on inside/outside test rather than selected joints.
- You may also create and animate a set of I.K objects first, then match any skeleton to them and click 'Bind Siblings' tool to set binding with a single click.
- For easy matching, radius can be controlled in edit mode via special purpose 'radius handle'.

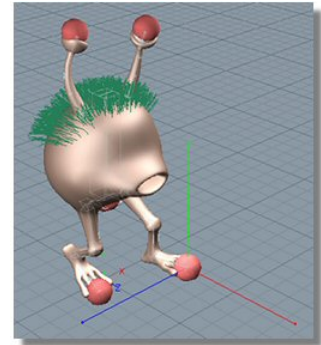
The tool is construction stack based so it can be combines with other type of animation effects. For example, use keyframing to fine tune the motions generated by I.K objects. Add noise to make the character to move and shake, etc.

Inverse Kinematics object defines the following attributes:

Position: Specifies the center point of the Inverse Kinematics object.

Radius: Radius of the object. This can be changed in the property window or by editing a special purpose radius handle in edit mode.

Snap To: If set, the joint is placed exactly on the center of the object. If not set, the initial displacement is maintained.

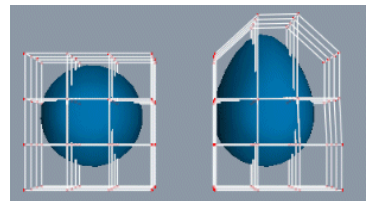


Five I.K objects attached to a body

Lattice Object

The Lattice object is a constructor based deformer. It is a three dimensional grid which is typically used for deforming freeform objects, such as SDS and NURBS surfaces.

The lattice object defines only attributes which specify the number of control points in each dimensions.



When a lattice object is created, the initial state of it is taken as the state of no deformation. Any modification applied to the lattice object will then deform the target objects accordingly.

The effect of editing a lattice object attached to an SDS sphere

Move Constructor

Move constructor translates the target objects based on the specified translation vector.

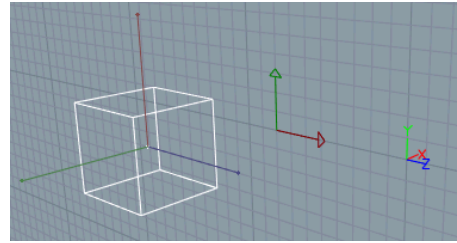
Packer

The Packer object is the key element for parametric object modeling. It arranges its child objects into a row and scales them to fill the space of the packer using a variety of options. 2D and 3D object arrays can be built by stacking packers hierarchically.

The property gadget of packer has the options listed below.

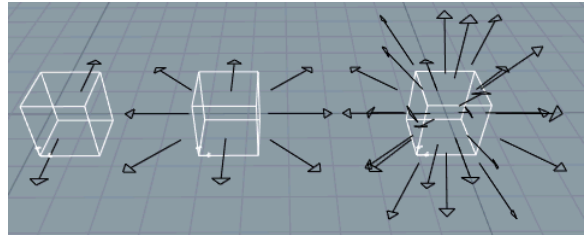
Drag And Drop Snapping options define what happens, when a packer object is dragged from the object browser window and dropped to a view window. The packer object has an ability to search closest other objects and snap into them. Snapping is based on collision detection. You can use the Hit Invisible property of geometric primitives to control and optimize snapping.

- Space: Snapping orientation.
 - Object: Snapping search directions (see Dir1, Dir2 and Dir3 below) are defined in the object space of the packer.
 - World: Snapping directions are defined in world space.
 - IPlane: Snapping directions are defined in the current input plane of the view window.



From the left: Object, Input Plane and World spaces

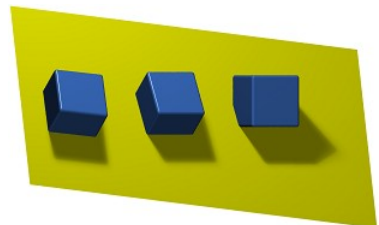
- Dir1: The primary snapping direction. Snapping can move the packer to the direction of a fixed axis, in a 2D plane with a given density (see Sampling below), or in all directions. Note that snapping directions are examined starting from the pivot point of the packer; this is sometimes a relevant factor. Setting Dir1=None disables the snapping functionality (other directions are not considered).



From the left: Z, XZ and XYZ(=all) snapping directions

- Dir2: The optional secondary snapping direction, which takes place after the primary snapping has been processed. For example, a kitchen wall cabinet can first snap to a nearest wall and then move side by side to a closest other cabinet. Note that if snapping space is set to Object, the secondary snapping happens in already rotated object space (see the Rotate option below).
- Dir3: The optional third snapping direction. For example, a kitchen cabinet designed to stand on a floor can first snap to the closest wall, then snap to the floor level and finally snap to the nearest other cabinet.
- Rotate: The packer can be rotated to match the surface tangent at the snapping position. For example, a kitchen cabinet should turn its backside against the wall to which it snaps. Rotate options define the 'route' of rotation as follows:

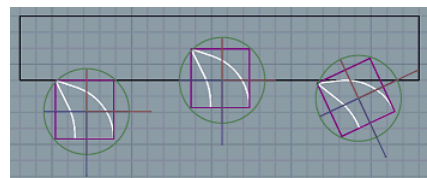
- Direct: The packer axis defined using the Axis option below is rotated to match the surface tangent using a single (shortest route) rotation. If the target surface has unusual orientation, this may tilt the object.
- H, P: The object is first rotated around its vertical Y axis (Heading angle) and then around the axis perpendicular to the selected fitting axis (Pitching angle). Two-step rotation eliminates tilting (i.e. Banking angle remains zero).
- H: The packer is rotated only around the Y axis.



Three cubes snapped to a tilted plane using Direct, HP and H (=heading only) rotation

- Axis: The axis which is rotated to match the surface tangent. The 8 main axis directions +-X, +-Y, +-Z of the selected snapping space are available. The option Any selects the axis that matches the tangent initially best i.e. selects the minimal required rotation automatically.
- Fit: Defines what is snapped to the found target point.

- BBox: The bounding box of the packer is fitted to the snap target point.
- Pivot: The pivot point of the packer is moved to the snap target point.
- Surface: The surface of the child objects of the packer is fitted to the snap target point using bi-directional collision detection.



Bounding box, Pivot and Surface fitting

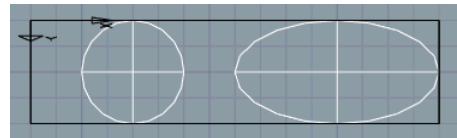
- Range: A distance threshold for snapping. Surfaces further than this are not considered in snapping.
- Sampling: Defines how many directions are examined in 2D and 3D snapping cases. For example, if Dir1=XZ and sampling is at the minimal level 1, four directions X, X-, Z and Z- are examined. If level is 2, 8 directions at 45-degree intervals are examined.

Geometry Managing options:

- Width: The current width (x dimension) of the packer object
- Height: The height (y dimension) of the packer
- Depth: The depth (z dimension) of the packer
- Min. Width, Min. Height, Min. Depth: You can set a minimal size for the packer object using these controls. For example, a kitchen cabinet assembled from 16 mm plate must be at least 32 mm wide (in practice at least 10 cm wide). The minimal size is initialized to the size of the packer at its creation time.
- Mod.X, Mod.Y, Mod.Z: ??

Packing options:

- Orientation: Defines the direction in which the packer arranges its child objects.
- Target: Selects a child object whose packing is controlled using the options below. If target=All, you can set packing options to all child objects at once.
- Expand: Expand the packing space reserved for the selected target to fill the packer.
- Unmanage: Disables packer management for the selected target. For example, material map objects are usually unmanaged children.
- Fill X, Fill Y, Fill Z: The selected target is stretched in the selected direction X, Y and Z to fill the packing space reserved for it.
- Pad X, Pad Y, Pad Z: Adds extra padding space around the selected target. The padding space is not filled even when the respective Fill option is enabled.



Two spherical objects packed. Both have Expand enabled and hence occupy an equal amount of space. The sphere at the right side has also Fill X enabled.

- IPad X, IPad Y, IPad Z: Expands the packing space reserved for the selected target.
- Anchor: The position of the selected target inside the packing space. Not relevant when the packing space is as large as the size of the target (Expand disabled or Fill options enabled to fill a dimension)

Path Level

Path Level object can be used for bending geometric objects using a nurbs curve, or any other geometric object. The actual geometric object defining the deformation is created inside the path level object.

Path Level can also be used to create path animations, by setting the Animator option, or by applying it to a moving object.

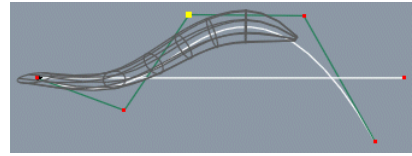
If the Animator option is not set, then the object uses the defined axis to represent non-deformed initial state and the defined curve to define deformed state.

If the Animator option is set, then the object uses the first defined curve segment to represent the deformed state and the rest of the curve is used for generating the path animation effect.

To create a wriggling snake, just apply the tool to a snake object with the Animator option set, or:

1. Create a horizontal cylinder mesh representing a snake.
2. Animate the snake to move horizontally (move the time slider of the last frame and translate the object in anim. record mode).
3. Activate the Path Level tool, make sure Animator option is not set and define a the first two curve points along the tube. Play the animation to see a wriggling snake.

A suitable banking option can be selected through the property window.



Point Translator

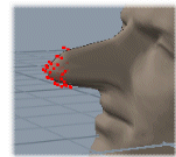
Point Translator is a constructor based deformer which can refer to number of geometric points in its target objects.

It can be used, for example, to define facial expressions.

The point translator object can be modified or animated similar to any other geometric object, in which case the effect is applied into the actual target points.

To use the point translator:

1. Select desired target points and click the Point Translator tool.
2. Activate the Construct option for the object and modify the translator object. The associated target points are modified accordingly.



Moving a point translator which deforms a nose

The Point Translator object defines Weight attribute, which can be modified in the Spec tab of the Property Window.

A point translator object can refer to any number of objects and points.

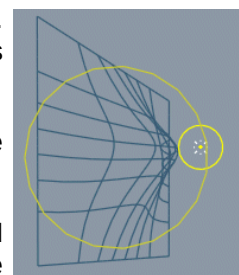
Radial Deformator

Radial deformator is a constructor based deformer. The shorter the distance to the center point of the deformer the stronger the deformation effect.

The initial position and radius of the deformer represents the state of no deformation. The desired deformation effect can then be achieved by editing the center and the radius of the sphere.

All points inside the initial radius will be influenced by the deformer. All points outside the sphere will not be affected by the deformer.

By moving the center point, one can achieve a magnet like effect. All points are moved with the sphere. The higher the distance of the point and the center of the initial sphere the smaller the translation.



A radial deformer applied to a nurbs mesh

To deform a nurbs mesh:

1. Create a nurbs mesh consisting of say 10 x 6 control points
2. Activate the radial deformer tool and specify two points defining the center point and radius attributes for the deformer.
3. Switch to edit mode and drag the center point or the radius handle of the deformer. The target points are deformed accordingly.

Setting the radius attribute to a smaller value generates a gravity like effect. In other words, target points are scaled towards the center point rather than translated with it.

The strength factor can also be controlled by defining a deformation curve through the property window. The default curve is linear. The beginning of the curve represents the maximum distance of influence and the end point corresponds to the zero distance.

The deformer also supports fractal noise deformation effect. The strength and density of the noise effect can be defined through the property window. This option is useful for creating turbulence effects. For example, a star ship can be moved through a noise field.

Rotate Constructor

Rotate constructor rotates the target objects about the specified axis.

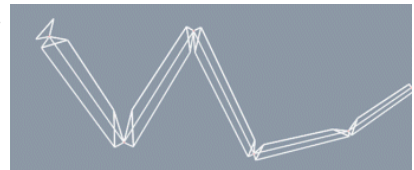
To rotate the selected objects by three full circles:

1. Click the Rotate constructor tool to activate it:
2. Enter two points in the view window to define rotation axis.
3. Specify the desired rotation angle through the property window. To make the target object to rotate three full cycles, enter $3 * 360$.

Skeleton

Skeleton is a deformer. Geometric points are bound to the joints of a skeleton through a set of binding tools.

In addition to this, it is also possible to use the lattice mapping system to attach desired objects to skeletons.



Skeleton properties are described below. For more information, see also Skeleton Tool documentation.

Default Joint Action

This option determines the action to be taken when the user drags a joint. The following options are available:

- Inverse Kinematics: Inverse Kinematics is used to move the selected joints. I.K propagates symmetrically to both directions translating joints according to their properties. Alt key can be used for enabling cumulative mode where the original joint positions are not restored in each mouse move.
- I.K Head, rigid tail: Inverse Kinematics propagates only to the joints at the head side. The tail side remains rigid.
- I.K Tail, rigid head: When set, Inverse Kinematics propagates only to sub joints. The head side remains rigid.
- Forward Kinematics: Applies forward kinematics to the joint. The dragged joint rotates about the previous joint.
- Move joints: The selected joints are translated, as if the skeleton was completely flexible. Angle constraints are expanded, if needed, to allow the defined translation.
- Modify lengths: Dragging a joint moves the joint closer/farther away to/from the previous joint.

Preferred Anim. Method

Defines the preferred method for key framing:

- Angles - the key framer records the bone angles for each joint.
- Inverse Kinematics: The key framer records the position of the selected joint only. When the animation is played back, the joint in question is positioned using Inverse Kinematics. Because of this, the state of the skeleton is not well defined, but depends on the initial state.

I.K Sampling

This option specifies the level of sampling for Inverse Kinematics. The higher the value the better I.K works.

I.K Parents

Specifies the number of parents I.K propagates to. If the value of this attribute is zero, I.K will never propagate to the parent object but affects only the current skeleton.

Mapping Volume

Mapping volume is a control hull with a circular cross section. It is used when one needs to bind a skeleton to an object to be deformed. Points inside the mapping volume will be bound to the skeleton and points outside the bounding volume will not be bound.

In hierarchical skeletons, mapping volumes typically overlap in which case certain point may get bound to more than one skeleton.

There are two options controlling mapping volumes:

- Mapping volume On/Off - to enable/disable mapping volume property
- Mapping Radius - radius of the mapping volume for the skeleton

Parent Joint

Skeletons can be grouped to a hierarchical tree where each skeleton may have one or more sub skeletons attached. For example, a hand skeleton may have five sub skeletons representing fingers.

This option specifies the parent joint to which the skeleton in question is attached to. Bone angles for such a skeleton are defined in the space of the parent joint.

Rotation from Parent

If this option is set, both the translation and the rotation of the skeleton is defined in the parent bone's space. If the option is not set, the rotation is defined independently from the orientation of the parent skeleton.

This option is effective only for skeletons attached to their parent skeletons via the 'Parent Joint' option.

Displacement

This option specifies how much the root joint of the skeleton is displaced from the joint of the parent skeleton. This option can be modified simply by moving the skeleton with the Move tool, or by defining displacement through the property window.

Interpolation

Realsoft 3D key framer supports two interpolation methods for key framing rotations.

- Angle Space
- Quaternion space

Angle space interpolation simply interpolates recorded rotation angles. The drawback of this method is that the two dimensional angle space has two singularity points. Any path passing near these points typically cause problems.

Quaternion interpolation solves this problem by using four dimensional angle space for interpolation. This is an advanced method and results very nice rotations. However, the problem is that the resulting animation curves are somewhat difficult to control.

Angle System

Skeletons support two kind of angle systems:

- HPB - Heading Pitching Banking
- HPT - Heading Pitching Twisting

HPB is the standard HPB angle system (stands for Heading, Pitching and Banking). The first two angles define polar coordinates for the bone. The third angle 'Banking' is defined so that as long as the 'x' axis of the bone lays on the ground plane, banking is said to be zero. This makes the method particularly suitable for controlling cameras. With zero banking, the x axes of the subsequent joints lay on the common plane.

HPT is a new angle system developed by Realsoft Graphics Oy.

The difference to the standard HPB system is that the third angle is defined in such a way that it minimizes the surface tension of the attached body rather than trying to keep the bones horizontally aligned.

Think about a striped flexible plastic pipe. Bend it to any direction and the stripes along the body tell you intuitively whether there are twisting or not. If the stripes appear spiral like, there is twisting involved. If stripes along the body follow straight lines, the twisting is zero.

In other words, when there is no real twist on the body attached to the skeleton, the third angle 'Twist' is said to be zero. What could be a more natural way of defining bone angles!

This system has several advantages over the standard HPB angle system. First, you typically have to edit only two bone angles: heading and pitching. You need to touch twisting angle only when you really need to get some real skin twisting introduced. In the standard angle systems, the banking angle has nothing to do with real twisting. In animations, you have to edit the banking angle all the time to eliminate unwanted twisting from the body. Another advantage is that I.K. can now operate based on heading and pitching angles only. This makes the I.K. tool much more useful.

The third and significant advantage of the new HPT angle system is that it helps to eliminate the biggest problem with angular spaces. In any angle space, there are two so called singularity points - the poles. When you animate, say, a camera over these poles, you'll get sudden jump at the north pole. The new HPT system eliminates this problem completely. In this respect, HPT angles are almost as good as Quaternion angles, and easier to handle in the curve editor.

Joint Angles

The orientation of each joint (or bone) can be controlled by three angles, named Heading, Pitching and Banking.

The joint angles are defined in the space of the previous joint. There is no previous joint for the root joint, in which case the angles are defined in the object space.

Skeleton has so called native orientation, in which the skeleton can be bent +/- 180 degrees. This corresponds to the 'a' (heading) angle of a bone. Joint representation visualizes this heading angle and intuitively shows you how much the joint in question can be bent.

In pitching angle, skeleton should only be bent less than +/- 90 degrees to avoid those problematic singularity points of the angle space. You can bend the skeleton beyond this limit, but in animations the skeleton may not behave as you expect. The reason for this is that the angle space has two so called singularity points at pitching = +/- 90. If you think the angle space as the surface of the earth, these singularity points correspond to the north and south poles. There, any latitude value will give you the same point (the pole). Therefore, you should try to create a skeleton

so that the direction in which it bends the most, corresponds to the heading angle. This problem is not specific to Realsoft Graphics Oy's implementation but is a fundamental problem related to polar angle systems.

If you need to create a joint which bends over ± 90 degrees in both directions, you may construct them from two joints close to each other. In fact, many real world joints are constructed this way. Another solution is to use quaternion angle system.

Constraint Angles

Constraint angles can be used for constraining bone angles. If a skeleton is bent over the constraints, the joint friction is internally increased preventing the joint from bending further and keeping the bone angles within the specified min / max constraint angles.

Note: If quaternion angles are used, constraints are effective only when interactively editing the skeleton. Constraints are disabled for animation play back.

Bone Length

Each joint defines a length attribute specifying the distance to the next joint.

Thickness

Thickness attribute specifies the thickness of the skeleton at the joint in question. This attribute is only used for visualization. It can be used for representing the actual body.

Position

Joint position attribute is actually an optional interface to joint angles. Skeleton can be defined by the root position and a set of joint angles, or by specifying joint positions for each joint.

Joint friction

The higher the friction, the more rigid the joint is when editing the skeleton using Inverse Kinematics. If the joint friction is 1, the joint is completely rigid and the two associated bones act as a single curved bone.

Fidelity

Fidelity attribute is based on multi weighted binding. If the fidelity is zero, then the each point is single bound to the bones. If the fidelity is 0.5, then the skeleton handles the binding as if the point was multi bound to subsequent bones.

This option specifies an object wise fidelity value. It is also possible to use point wise fidelities.

Drag

This attribute specifies the drag force between the joint and the medium. The higher the value, the stronger the joint resists attempts to translate it. Infinitely high drag force anchors the joint to a world.

Anchor

Anchor option anchors the joint coordinates in the world space or in the object space when the skeleton is modified using the Inverse Kinematics tool.

If the joint is anchored in the world space, the joint maintains its position even when the entire object is translated.

If the user modifies an anchored directly, the anchore is temporarily disabled for the joint.

Rigid

Setting this option makes the joint rigid. The effect is the same as setting joint friction to 1.

Animate as I.K Target

When this option is set, the key framer records the position coordinates. In animation play back, the joint is moved to the specified point using inverse kinematics.

Animate as Angle

If set, the key framer records bone angles for the joint. This option is typically used when Preferred animation method is set to Inverse Kinematics and when some of the joints need angle based key framing.

I.K Head Stopper

Inverse Kinematics stopper. When one of the joints at the tail side is modified using I.K, this option prevents I.K from propagating to parent bones. The option has no effects when I.K is applied to parent bones.

I.K Tail Stopper

Inverse Kinematics stopper. When one of the joints at the head side is modified, this option prevents I.K from propagating to parent bones. The option has no effects when I.K is applied to tail side bones.

Binding

Skeleton acts similar to other construction based deformaters. The main difference is that skeleton must be *bound* to target objects before it can deform any target points.

When the skeleton is bound, its current state is saved. When the skeleton is then modified, a deformation is computed from the difference between the initial state and the current state.

Binding becomes effective only when the *Construct* feature is enabled for the skeleton. Resetting *Construct* option does not discard binding data.

Two methods exist for setting up initial binding.

Single binding binds target points to their nearest joints of the skeleton.

Multi binding uses bounding volume property to determine the skeletons to which the points should be mapped.

It is also possible to set up or fine tune binding manually.

Bind Selected tool binds the selected points to the selected bones. The tool is shown when one target object and one skeleton object is selected. This tool does not set up multi binding. If multiple bones are selected from a single skeleton, point is mapped to the nearest one.

Unbind Selected unbinds the selected points from the selected bones. The tool is shown when one target object and one skeleton object is selected.

Unbind tool detaches all target objects from the skeleton. All binding specific data is freed from the skeleton.

Select tool allows the user to select the points bound to the selected joints. This tool selects all points bound to the currently selected bones. The tool is only shown when both the target object and the skeleton object are selected. The tool is useful, for example, when one wants to view the points bound to a certain bone. Or, when some of the points got bound to a wrong bone and you want to unbind them.

Unselect tool unselects all points bound to the selected bones. The tool is only shown when two objects are multi selected: the skeleton object and the target object. The tool is very useful when binding points manually to complex hierarchical skeletons because it allows the user to exclude points which are already mapped to other skeletons.

Handles

Skeleton supports a set of handles through which the user can control the skeleton.

Handles can be switched on / off through the Wire tab of the Property Window, or through the compass menu.

Joint Handles

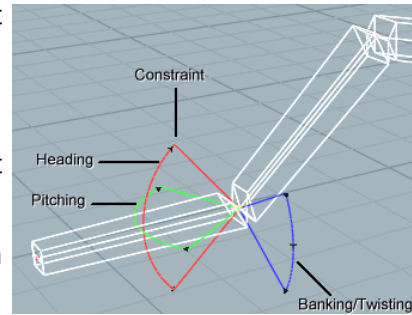
Each joint acts as a handle activating the tool specified by the 'Default Joint Action' option. The default tool is Inverse Kinematics.

Angle Handles

Angle handles can be used for controlling bone angles and constraint angles.

Dragging an angle handle modifies the corresponding bone angle, such as Heading, Pitching or Banking.

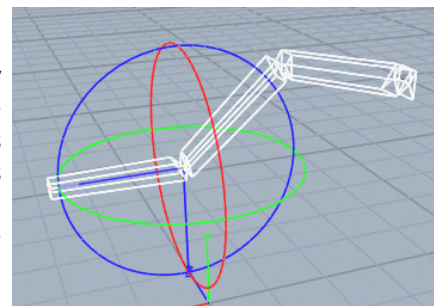
The constraint handles are marked as black arrow heads at the both ends of each angle handle.



Angle handles

Rotation Handle

The rotation handle rotates the selected bones exactly the same way as the standard rotation handles rotate regular geometric objects. The difference to the bone Angle Handles is that the rotation handle rotates the bone about the main axes of the bone whereas bone Angle Handles control certain bone angle directly. In order to get the desired rotation applied, rotation handles typically need to change all the three bone angles.



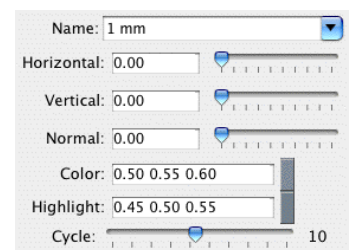
The rotation handle can be used for rotating the selected bone. Constraints can be disabled by holding down the Alt key.

Bone rotation handles

Grids

Grid object has the following properties:

- Name: The name of the grid object
- Horizontal, Vertical, Normal: sets the grid dimensions, which define the distance between snap positions. Usually, all three fields are set to the same value. If you wish to enter small values like 0,001, make sure the value of the options/window/metrics tab is set to match.
- Color: The base color used for grid drawing.
- Highlight: A second color used for drawing the grid lines, whose frequency is determined using the 'Cycle' slider.
- Cycle: Defines the density of grid lines that are drawn with the highlight color.



You can activate the desired grid by dragging and dropping it into the view window. This sets the current grid for the view window and activates both 'snap to grid' and 'show grid' options. You can also use view property window's 'Input' tab for controlling view's grid snap and draw options.



Note

Do not mix the grid with the ground plane visualization in perspective views. The ground plane has nothing to do with snapping and is provided as an orientation guide only. It is automatically turned off when grid drawing is enabled.

The view control bar allows you to switch snap to grid and show grid options on/off quickly any time.

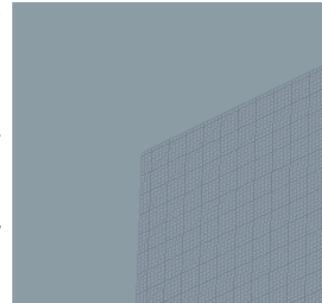


Draw Grid and Snap to Grid icons

The grid drawing fills the entire view window. If you zoom out the view window so that the grid appears denser and denser, the grid system stops drawing the base colored lines at some point, because the too densely drawn lines would not serve any useful purpose. Drawing only the highlighted lines keeps grid visualization useful. If you continue zooming out, also highlighted lines get too dense and their drawing will be stopped.

The same density control is used when perspective projection is enabled and view camera is not locked to the grid plane. Only the part of the infinite grid plane, which is not too dense, is visualized.

The automatic density control does not affect grid snapping, only grid drawing.



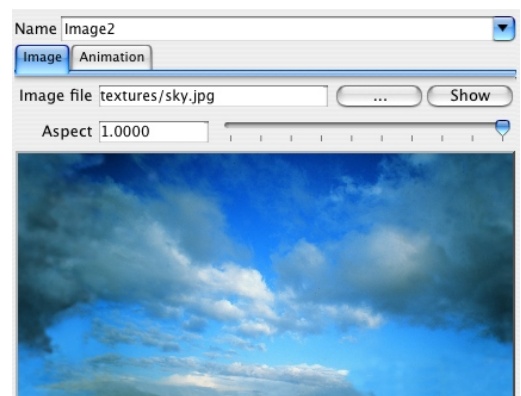
Too dense part of the grid is not drawn in perspective mode

Images

An image object displays a disk image file or a frame of an animation file (e.g. an AVI file). You can use it for viewing purposes. For example, when 3D painting an object, the system automatically adds an image object which displays the painting result in real time.

The image object properties are:

- **Name:** A symbolic name for the object.
- **Image file:** The disk file name. The path may be omitted if image search paths include the parent folder.
- **Aspect:** Pixel aspect ratio height/width. With this control, you can fix stretching of images coming from devices, which have non-square pixels (such as popular TV standards PAL and NTSC).
- **Animation:** Frame index controls. With these, you can activate automatic image updating by animation time. For example, if you want to synchronize a 3D animation to a live video stream, just create an image object that uses an avi file or a frame sequence, activate Automatic animation indexing from the Animation tab and drag&drop the image object to a view window. The view window backdrop will automatically update whenever you change the animation frame. For detailed description of these controls, see the Animation tab documentation.



Materials

Material Types

Realsoft 3D has an open architecture, which means that a material object can have any kind of property or interface. This chapter gives an overview of the material types included in the program. You can find the collection of installed material classes by opening the 'New' sub menu from the popup menu of the select window's material section.

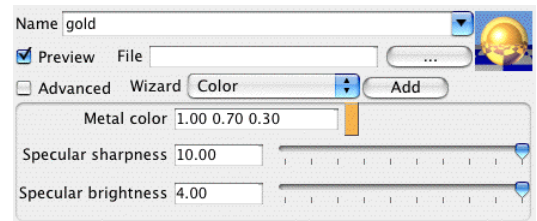
VSL Material

This is the most powerful and commonly used material type of the program. All materials, which you can find in the default material library, are of this type. The reason why VSL materials are so powerful is that the 'low level' interface of the VSL materials is a programming language. Advanced users can therefore implement unusual and customized material effects. Don't be alarmed by the term 'programming language', you do not have to write complex formulas and algorithms, because VSL materials can be built using wizards and an easy drag and drop interface. Consequently, it is easy to get started. Nevertheless, in order to fully utilize the possibilities of VSL, some programming experience is required.

Using VSL materials is very easy. VSL makes it possible to build an optimal interface for each material type. You can see this by examining the materials of the default library in the property window.

The following control groups are available for VSL material building and editing:

- Top Level Controls
- Advanced Controls
- Object Specific Tab
- General Tab
- In/Out Tab



A VSL material in showing its 'easy to use' interface

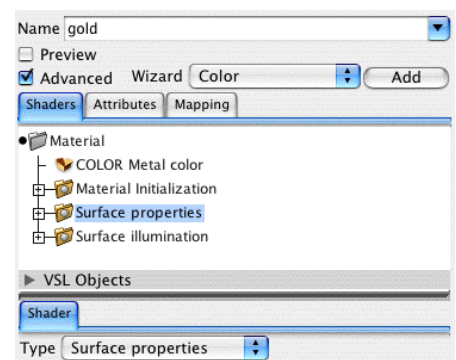
Top Level Controls

The topmost area of the VSL material editor shows a set of gadgets that are displayed both in the easy to use interface and in the advanced editing state. These controls are:

Name: The name of the currently edited material. The editing target can be changed by choosing another material from the list.

Preview: This checkbox enables/disables the material preview feature. When enabled, the program automatically renders a test image of the edited material after each change. The preview is rendered using a test scene defined by the current material preview project. It is also possible to assign a special preview project to the material by selecting a disk project file using the 'File' gadget at the right side of the 'Preview' check box. The rendered preview image automatically updates the material icon in the select window (if preview icon feature is enabled).

File: A custom preview project. If a project file name has been defined to this field, the program automatically loads the project and uses it for material previewing whenever the material is edited in the preview state. The idea is that some materials may require a certain kind of preview project which shows the characteristics of the material well. For example, a transparent material requires some objects behind it so that refraction can be tested. The 'previews' folder in the Realsoft 3D home directory contains some predefined preview projects. You can create more of them by editing the default material preview project and saving the result to the 'previews' folder with a suitable name.



A VSL material in the advanced editing state

Preview Window: The squared little window at the top right edge of the material editor is actually a View Window. Therefore, you can rotate, zoom and pan it with the right mouse button and Alt, Shift and Apple modifier keys. The window also has a small popup menu:

- *Reset:* Resets the viewing position back to the default front view.
- *light properties, material preview, ...:* The rest of the menu items show the contents of the 'preview' folder. You can choose a suitable preview project here instead of browsing from the 'File' gadget.

Advanced: This checkbox brings up the advanced VSL editing controls.

Wizard: A list of VSL wizards, which automatically add some commonly needed material appearances and features to the edited material. For more information, check the wizard list.

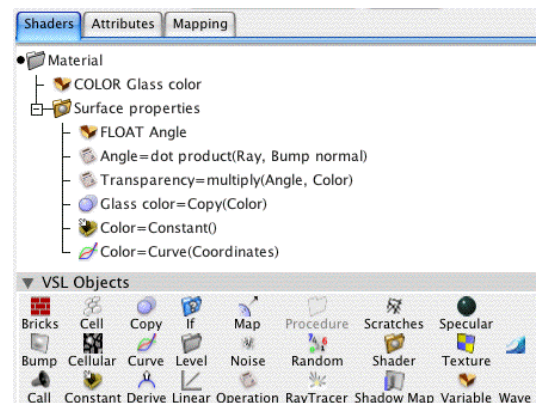
Add: Adds the currently selected wizard elements to the edited material. As a result, a set of new VSL objects and top level controls appear to the material. Note that the same wizard may be applied several times to the material; the program does not check or prevent this. For example, a material may include several texture maps which are blended together.

Advanced Controls

The advanced VSL controls are divided into the following three tabs:

Shaders Tab

Shader Tree - The tree diagram shows the complete structure of the 'shading program'. The top level objects, 'Shaders', are executed in a predefined fixed order, regardless of their order in the constructed VSL tree. Inside the shader levels, the execution proceeds in top-down order. Variables can be placed freely into levels - position does not matter - but is a good programming practice to introduce them as the first objects in a level. VSL objects can be inserted and moved around using drag and drop. Apple-dragging creates a duplicate and moves it. The shader tree gadget has the following popup menu (it is context sensitive and changes depending on the VSL object selection):



- *New* - Contains a list of available VSL objects. Selecting a sub menu item adds the corresponding VSL object to the currently edited VSL level.
- *Cut* - Cuts the selected VSL object (and its possible sub objects) into Realsoft 3D's VSL clipboard.
- *Copy* - Copies the selected VSL object into Realsoft 3D's VSL clipboard.
- *Paste* - Adds the contents of the Realsoft 3D VSL clipboard to the end of the edited VSL block.
- *Duplicate* - Duplicates the selected VSL object to the end of the same level.
- *Delete* - Removes the selected VSL object.
- *Output* - You can select the output channel/variable of the selected VSL object from this menu.
- *Operator* - The mathematical operation which is used when assigning the result of the evaluation of the selected VSL object to its output parameter.
- *Input0, Input1 ...* - You can set the input parameters for the selected VSL object with these menus.

- **Rename** - Renames the selected VSL object (slow double click also works). VSL objects can be renamed to make the code better understandable for other users.



Note

Renamed objects no longer update their names automatically when the input and output parameters change.

- **Disabled** - Removes the selected VSL object from rendering computations. This feature is useful when building and experimenting with the VSL code.
- **Balanced Layout** - The VSL hierarchy will be displayed using horizontal tree layout.
- **Properties** - Opens a floating window which shows the object specific attributes of the selected VSL object.

VSL Objects - An array of currently installed VSL objects. You can drag and drop items from this array to the VSL shader hierarchy. Clicking a VSL object icon adds it to the end of the level where the currently selected VSL object is located. Note that VSL object hierarchy follows some syntax rules, and a certain object may not fit into some parts of the hierarchy. For example, the root level of the hierarchy can contain only variables and shaders, and a shader cannot be placed into another shader. When using drag and drop, the mouse pointer symbol tells you where you can drop an object. If you click a VSL icon in the array and nothing happens, the reason is most likely that the current destination is not allowed.

Attributes Tab

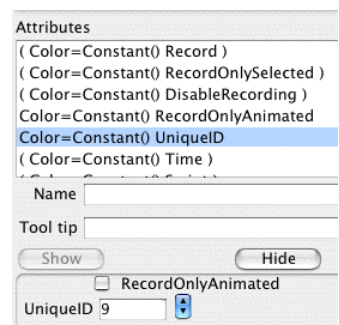
The Attributes tab contains the tools for creating an easy to use interface for VSL materials. The controls are:

Attributes - A list of all attributes of all VSL objects in the current material. You can pick an item from this list and make it visible in the easy interface with a suitable name. The list includes a simple popup menu, whose items 'Show' and 'Hide' serve the same purpose as the two buttons described below.

Name - Displays the name of the selected attribute. By default, attribute names describe their general nature. For example, 'Amplitude' defines the magnitude of the 'Noise' object's value variation. You can change the name to describe the particular purpose of the attribute in the edited material. For example, 'Wave Height' might be a better name than 'Amplitude' for a sea water material.

Show - Adds the currently selected attribute to the easy interface.

Hide - Removes the currently selected attribute from the easy interface.



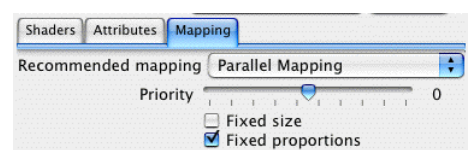
Note

Attributes of deleted VSL objects automatically disappear from the constructed interface.

At the bottom of the Attributes tab, you can see the defined GUI for the material. The same interface will become visible when you turn off the advanced editing state.

Mapping Tab

The rightmost tab of advanced controls defines how the created material should be mapped to target objects. The specifications made here are used by the drag & drop system. The user can overrule them by selecting another mapping type from the mapping creation tools.



The following controls are included:

- **Recommended Mapping:** Select the preferred mapping type from this gadget.
- **Priority:** Evaluation order recommendation. By default, materials are evaluated in the order they appear in the hierarchy, deep sub level materials last. The hierarchy defined evaluation order can be overruled with the mapping priority setting. The higher the priority, the earlier the material will be evaluated.



Note

This means that the material will have a weaker effect! Later evaluated materials can overwrite the result. Some special materials, such as a global illumination shader, may work best if it is evaluated last. Because most materials have the default priority 0 and only some built in shaders use evaluation priority -2 .. +2, a value -5 is usually sufficient to guarantee that the material is evaluated last.

- **Fixed size:** If set, the size of the mapping geometry does not automatically resize to match the size of the mapping target object.
- **Fixed proportions:** If set, the proportions (width, height and depth) of the mapping geometry do not automatically stretch to match the size of the target object. For example, parallel map always becomes a symmetric cube if this option is set.

If the recommended mapping is set to 'Any Mapping', drag & drop system creates a mapping that best matches the geometry of the target object. Spheres will get a spherical mapping, cylinders gets cylindrical mapping, etc.

Object Specific Tab

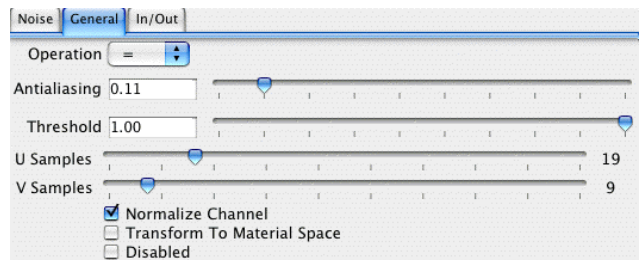
The leftmost tab on the bottom of the VSL editor window shows the object specific attributes of the selected VSL object. Therefore, the tab contents changes from object to another. Object specific VSL attributes are described in VSL objects.

General Tab Options

The majority of objects have several common controls, shown in the 'general' tab of the VSL editor.

Operation:

This option defines how the value computed by the object is assigned to the output channel/variable. If the output channel has 3 subchannels, the same assignment is considered for each sub channel. Operation can also be changed using the popup menu of the VSL editor.



=	Default assignment, overwrites the old value with the new one
+	Adds the new value to the old value
-	Subtracts the new value from the old value
*	Multiplies the old and the new value
/	Divides the old value by the new one
Min	Assigns the minimum of the new and the old value to the output channel

Max	Assigns the maximum of the new and the old value to the output channel
Abs	Assigns the absolute value of the computed expression to the output channel

Normalize channel:

This option is provided for convenience. Many of the available channels are such that they must be normalized whenever their value is changed. The term 'normalize' means that the value of the channel is divided by its length, the result being always a unit vector. For example: normalization of vector (3, 4, 0) equals (3/5, 4/5, 0). The channels, which are (and should be kept) in normalized form are:

- Surface normal
- Bump normal
- Ray
- Reflected ray
- Refracted ray

Transform to material space:

This option performs a 3D transformation to the orientation defined by map x, map y and map z channels

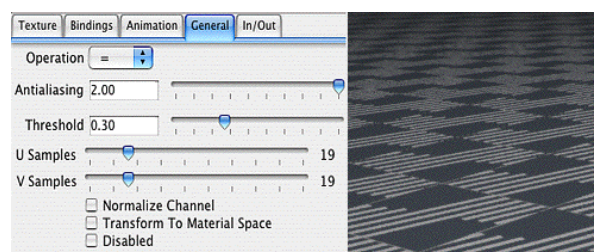
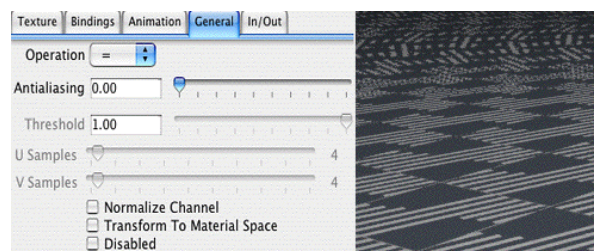
$(x,y,z) \rightarrow x \cdot \text{map } x + y \cdot \text{map } y + z \cdot \text{map } z$

The orientation defined by these vectors is often suitable for surface normal modifications, such as bump mapping.

Antialiasing, Threshold, U-samples, V-samples: These settings control material anti-aliasing. Generally speaking, anti-aliasing is a method of increasing computation accuracy because of high 'information density' or 'level of detail'. In other words, if an examined pixel contains plenty of details, the rendering system needs to analyze that point carefully to achieve proper image quality.

Antialiasing defines the relation of how quickly computation accuracy is increased when level of detail grows. If the value is zero, anti-aliasing computations are switched off. A good default value is 1.

The Threshold setting controls the level of detail when the anti-aliasing starts. The lower the value, the easier antialiasing starts. The minimum value is 0.



The effect of VSL anti-aliasing in a texture map. The quality of the first picture (no anti-aliasing) is not sufficient for animations because of intolerable amount of flickering. The second picture shows the improved quality after activating the anti-aliasing feature.

U-samples and V-samples define how many times the object is maximally evaluated for anti-aliasing purposes. The system is designed so that by setting the u & v samples to the level that corresponds the 'native' level of detail of

the object (for example: pixel resolution of texture map object), anti-aliasing factor 1 and threshold value of 1 are optimal. However, because of speed reasons, it is not always possible to use high enough sample values (1000000 samples for a 1000*1000 pixel texture map is too slow), so sometimes lowering the trigger value and increasing the antialiasing value may be necessary.

Note also that sometimes it may be necessary to use higher u-v-sample values than the texture resolution: for example, a simple 2*2 pixel checker pattern may require 4*4 samples because of the random nature of the anti-aliasing process.

Disabled

This option disables the object in ray trace rendering. The result is the same as if the object were deleted. The option is useful when experimenting with the materials.

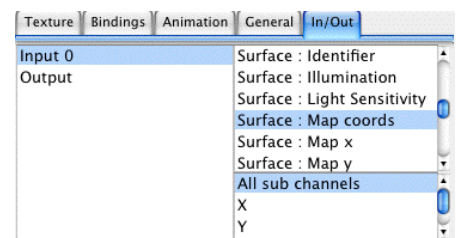
Tip

The Tip string can contain some optional information about the purpose of the VSL object, instructions how to adjust the shader or other user-defined information. The Tip string control is the preferred way to add such comments, because renaming a VSL object stops the automatic name updating when input or output parameters or other attributes affecting the name of the VSL object change.

In/Out Tab

VSL object parametrization can be controlled using the 'In/Out' tab of the property window.

The list on the left side shows all input and output parameters of the selected VSL object. The list on the top right side shows how the parameter is assigned. The list includes all channels and variables available in the current shader.



The third list is for sub channel selection. The following examples illuminate this feature:

- Parametrization by all sub channels: `myparam=linear(map coords)`
- Customized sub channel parametrization: `myparam.x=linear(map coords.y)`

VSL Procedure Library

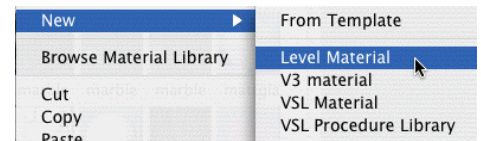
A VSL procedure library is an object which can contain a set of general purpose VSL functions. For example, an advanced user may create a VSL procedure library called 'Noise Functions' and place frequently needed noise related VSL algorithms there.

When a procedure library is selected in the VSL editor, only Procedure objects can be added at the top level of the VSL hierarchy. Each procedure defines a symbolic name, a parameter list and some VSL code. Note that VSL objects in a procedure can refer only to the defined input parameters and to the output value (procedure's return value). This supports good programming. All required data must be passed in as parameters. Input parameters are used by address, not by value, so assigning a value to an input will change the original parameter defined in the call. This also means that a procedure can return multiple output values.

All defined procedures in all memory-resident procedure libraries can be called from the usual VSL materials. A procedure call is defined using the Call VSL object. A call to a non-existing (unloaded or deleted) procedure does nothing.

Level Material

The Level material allows you to combine existing materials to construct more complex materials. Thanks to this, you can map several materials to an object using a single mapping object. The tutorial 'Blending Materials' already showed one typical use of the level material: combining a material and a 'scope mask', which defines where the material has influence.

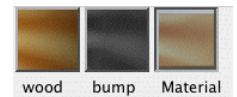


Create a Level Material

Creating level materials

Level materials can be created from the select window's New menu (when the Materials tab is selected).

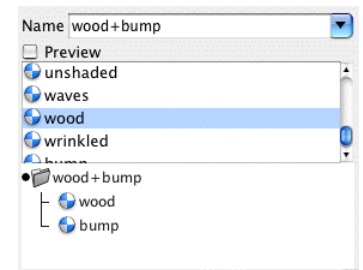
This material can refer to any number of other materials. The sub materials of a level material can be selected simply by dragging the desired materials into the level material in question.



Wood and bump are dragged and dropped into a level material. The result is a bumpy wood material

Modifying level materials

Level materials can be edited as usual: just double click the desired level material in the select window and the property window is opened allowing you to edit the material. Also the property window supports drag&drop. Drag the desired materials from the upper material list into the level material.



A level material consists of two sub materials: wood and bump

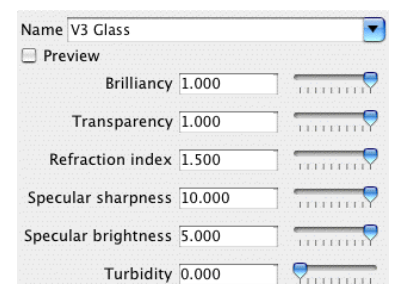
V3 Material

This material type implements the same kind of controls for optical material properties as Real3D v.3. 'Brilliancy' property defines the amount of mirror like reflection. The amount of diffuse shading is automatically computed from brilliancy: if brilliancy is zero, diffuse shading is at its maximum and vice versa. The material can be used for basic mirror, glass, etc. and it renders quickly.



Note

The more general VSL material allows all effects that 'V3 material' can define, but V3 material is slightly faster to render.



Samples

The channels are not used 'one by one' in the rendering engine; instead, they are managed as sets of channels. One such set contains space to store all the available channels. A channel set is called a 'sample'. Because all channels are included in a sample, it can store all attributes of an object in a point of space.

One sample is not enough to store the data for the entire shading process. In order to fully utilize the capabilities of the shading system of Realsoft 3D, one must know the relation between the sample objects the rendering engine uses and the shaders.

The sample objects are named according to the purpose they are used for. In some shaders, only one sample is available; some shaders provide two samples (perhaps even more in future releases or plugins for the software). For example, in the surface illumination phase, the first sample describes the properties of the surface, and the second sample describes the properties of the light. In such a situation, the channel names are prefixed by a name that specifies to which sample the channel belongs. For example, Light:illumination is the channel that contains the color of the light from a light source, and Surface:illumination is the channel that contains the total color emitted from the surface.

Below is a list of sample objects the rendering engine uses to carry out the shading process.

Light sample

This sample contains the properties of a light ray (from a light source). The sample is used first in the light properties shader, next in the surface and volume filtering shaders and finally in the surface/volume illumination shaders.

Surface sample

The most frequently used sample. Describes the properties of a surface (boundary of a geometry). The surface sample is available in many shaders and it is used to carry the result information through the rendering system. The ray tracer computes its default shading effects to the illumination channel of the surface sample. It contains the 'result' color of the rendering process at the end of the ray tracing.

Surface filter sample

This sample is used when examining shadow casting properties of a surface in the surface filtering shader.

Particle sample

The particle sample is used to examine properties of particle objects that are drawn in post processing. The sample is used only in the post particle phase.

Ray hit sample

This sample is used with ray-surface hit events. The main purpose is to modify channels that affect the shape of the surface. The most important of such channels is the distance channel. This sample is used only in the ray hit shader.

Volume sample

The Volume sample is used to examine properties inside objects (the surface sample is used on the boundary). It is used first in the volume properties shader, then in the volume illumination and volume shading shaders.

Volume filter sample

This sample is used when examining shadow casting properties of a volume in volume filtering shaders.

Shaders

The highest level of a VSL material hierarchy consists of Shaders. A Shader is a VSL object that has only one property: 'Type'. 'Type' has values like 'Surface properties', 'Surface illumination' and 'Volume properties'. These names, quite obviously, reflect the actions that the rendering engine performs.

The order in which the shaders appear in the hierarchy is not significant. For example, the rendering engine needs to know surface properties before it starts computing surface illumination, and therefore the corresponding shaders are executed in this order, although the user may have inserted them in reversed order.

The set of channels, which is available for use in each shader type, varies. This helps to understand when channels become initialized and which channels are sensible to change. For example, in the 'Volume properties' shader, the

surface normal channel is not available at all. This is because it does not make sense to deal with a surface attribute inside an object. In this way, the shader structure guides the user and helps to understand the rendering process.

The shader system is expandable. Plugin modules may insert new shader types.

The default shader types are listed below. These shaders describe the steps of the rendering process quite accurately, so rendering can be customized quite thoroughly using the shaders. For example, the default surface lighting model (standard diffuse lighting) can be redefined completely using suitable objects in the 'Surface illumination' shader.

Included is also a list of 'system' channels that are initialized by the rendering system particularly for that shader, or by an object, which is typically evaluated before the shader. In addition to these, other channels may also contain initialized information written by user defined special operations, materials, plugin classes or by other means.

Material Initialization

This is a special shader in that it is executed only once; at the beginning of rendering. Therefore, it is suitable for initializing constant data, which is stored in root level variables.

Initialized channels:

Time	The global 0..1 time

Common Properties

The VSL code included in this shader is always executed with all other shaders of the material. You can place general-purpose operations and preparations that affect all other shaders into this shader. A typical use of this shader is defining the scope of a material - how much the material affects the target object.

Note, that the position of the common properties shader in the VSL hierarchy defines its evaluation time: if it is placed above other shaders, it is evaluated first. If it is at the end of the VSL tree, the code is evaluated as the last action of every other shader.

If the material contains no other shaders as the common properties shader, the evaluation defaults to the surface properties shader.

Initialized channels:

Scope	The channel defining how strongly the material affects the targets.
Map coords	Mapping coordinates is the usual parameter for texture maps, etc.
Time	The global 0..1 time.

Output channels:

Map coords	You can compute a global change of mapping coordinates in this shader.
Scope	Scope defined in this shader affects all other shaders.

Surface Geometry

This shader can be used to modify surface geometry related attributes.

Initialized channels:

Ray	The direction from which the surface point is looked from.
Distance	The distance from the ray source (camera, etc.) to the surface.
Coordinates	The point of the surface being examined.
Time	The global time.
Normal, U direction, V direction.	Local surface orientation
Map coords	Mapping coordinates.
Map x, Map y, Map z	Mapping orientation.
Antialiasing	Also computed by most mapping objects.

Channels used by the rendering system after the shader:

Bump height	Modifying this channel creates bump map effects. The height is expressed as meters. Therefore the values are typically quite small.

Surface Properties

This is probably the most commonly needed shader, because the appearance of most objects is defined by their surface. Normally, properties such as color or transparency are defined in this shader.

Initialized channels:

Ray	The direction from which the surface point is looked from.
Distance	The distance from the ray source (camera, etc.) to the surface.
Coordinates	The point of the surface being examined.
Color	The default diffuse color (white) or object's color.
Time	The global time.
Normal, U direction, V direction	Local surface orientation.
Bump normal	Possibly modified (bumped) surface normal.
UV Coords	Local surface coordinates.
Map coords	Mapping coordinates.
Antialiasing	Information density computed by most mapping objects.
Map x, Map y, Map z	Mapping orientation.
All 'Exterior' sample channels that are initialized in the 'Volume properties' shader.	These channels describe the properties of the volume from which the ray hitting the surface arrives. For example, Optical thickness and user defined channels can be utilized for optical effects. For more information, see 'Volume properties' shader below.

Channels used by the rendering system after the shader:

Color	The value defines diffuse shading.
Bump normal	Modifying this channel creates bump map effects.

Ray	Used in various shading actions.
Reflection	Amount of mirror reflection.
Transparency	Amount of transparency.
Volume sampling	Values greater than or equal to one invoke volume sampling.

Volume Properties

This shader is used when examining interiors of objects. The shader is needed when creating fog effects, including volumetric light effects (fog glowing in light, shadows casted in fog). Volume sampling slows down rendering, and therefore there is a switch on render settings, 'Volume sampling', which must be enabled in order to let volume properties shaders to operate.

Initialized channels:

Volume:Distance	The distance between samples (=diameter of space area the sample represents).
Volume:Coordinates	The examined point.
Volume:Ray	The direction of the examined ray.
Volume:Time	The global time
Volume:Map coords, Volume:Map x, Map y, Map z, Volume:Antialiasing	The mapping specific information. See 'Surface properties' above.
Ray:Distance	The total distance ray travels in the volume.
Ray:Illumination	The result of ray tracing so far.

Channels used by the rendering system after the shader:

Volume:Color	The diffuse color of fog. This property reacts with lighting.
Volume:Illumination	The 'ambient' or 'self-illumination' color of fog.
Volume:Turbidity	Density of the fog. Defines overall visibility of volume effects as a function of distance.
Ray:Illumination	It is possible to leave the 'Turbidity' channel to zero and compute the volumetric effect directly to this channel.

Light Properties

This shader defines the properties of light emitted from a light source. For example, it is possible to define light falloff rate, color, additional light property channels (a light source casting gamma rays), etc.

Initialized channels (all in Light sample):

Distance	The distance from the light source to the illuminated point.
Coordinates	The illuminated point (NOT the position of a light source!).
Ray	The direction of the light ray (from the light source to the illuminated point).
Illumination	The color of light.
Time	The global time.

Map coords, Map x & y, Antialiasing	Mapping information (see 'Surface properties' above).

Channels used by the rendering system after the shader:

Illumination	The color of light. If black, no illumination is computed.
Distance, Coordinates, Ray	These will be used in shadow computations.

Surface Filtering

This shader controls the shadow creation. A ray tracer executes the shader when a light ray coming from a light source hits an object. The purpose is to find out if the material applied to the object lets the light go through and how the light changes. The decision is made by examining the transparency channel. If the color is white (1.0, 1.0, 1.0), the light continues in an unmodified state, whereas black (0,0,0) stops the light and causes a full shadow. Materials can define custom filter effects by computing suitable filter color values in the filtering shader.

A Ray tracer always executes the 'Surface properties' shader before 'Surface filtering' shader. Therefore, transparency defined in 'Surface properties' is conveniently used as a default value for 'Surface filtering'.

Initialized channels:

All Light sample channels that were initialized in the Light properties shader	See 'Light properties' above.
All channels of the Filter sample that were initialized in the Surface properties shader	See 'Surface properties' above.
Light:Color	Unshadowed, original amount of light.
Light:Illumination	The current amount of light (possibly reduced by other shadows).
Target:Antialiasing, Target:Distance etc.	Target refers to the surface receiving the shadow. Some channels initialized in the Surface properties shader of the shadow Target are available mainly for advanced antialiasing purposes.

Channels used by the rendering system after the shader:

Light:Illumination	The current amount of light. You can change it directly for custom effects.	
Filter:Transparency	The rendering engine computes the amount of shadow from this channel. White = No shadow.	
Filter:Fade	Also the fade channel contributes to the shadow transparency. Fully faded objects do not cast a shadow.	

Volume Filtering

This shader is executed to examine how a volumetric material shadows the light. The sampling density channel defines the accuracy of computation, just like in other volume effects. Furthermore, the 'volume shadows' option in render settings needs to be switched on. The transparency channel defines the amount of shadow. Transparency has the default value defined in the 'Surface properties' shader, which was evaluated when the light ray entered the volume.

Initialized channels:

All Light sample channels initialized in the Light properties shader	See 'light properties' above.
Light:Color	Unshadowed, original amount of light.
Light:Illumination	The current amount of light (possibly reduced by shadows).
VFilter:Coordinates	The examined point.
VFilter:Distance	The distance between samples (=diameter of space area the sample represents).
VFilter:Map coords, VFilter:Map x, Map y, Map z, VFilter:Antialiasing	Mapping specific information.
VFilter:Ray	The direction of the examined ray.
VFilter:Time	The global time.
VFilter:Transparency	Initialized to the value defined by Surface properties.

Channels used by the rendering system after the shader:

Light:Illumination	The color of light. Can also be modified directly.
VFilter:Transparency	The rendering engine computes the amount of shadow from this channel.

Begin Illumination

This shader starts processing surface illumination by light sources. It is executed once, and after that usually multiple Surface illumination evaluations follow. In this shader, you can initialize channels for collecting lighting information.

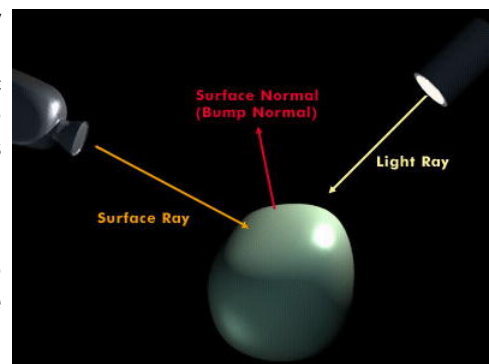
Initialized channels:

All Surface sample channels that were initialized in the Surface properties shader	See 'Surface properties' above.

Surface Illumination

This shader is executed for examining the influence of a light ray to a surface. This is a suitable shader for defining custom lighting effects. It is possible and also common to define illumination specific information already in the 'surface properties' shader, such as the usual diffuse color. Nevertheless, at this shader, the properties of the light ray are available (intensity, color, direction, etc.), and therefore, more detailed and advanced effects can be defined.

The default actions after the surface illumination is computing the diffuse shading using the color channel and adding the result to the illumination channel, which describes the total result of the shading.



Surface illumination

Initialized channels:

All Light sample channels that were initialized in the Light properties shader	See 'Light properties' above.

All Surface sample channels that were initialized in the Surface properties shader	See 'Surface properties' above.
Light:Color	The original amount of light sent by a light source.
Light:Illumination	The amount of light which reached the surface.

Channels used by the rendering system after the shader:

Light:Illumination	The amount of light.
Surface:Color	Diffuse lighting depends on this channel.
Surface:Illumination	The total result of rendering. Add custom illumination to it.

End Illumination

This shader finishes processing surface illumination by light sources. You can examine and store the total influence of light sources to the surface in this shader. For example:

```
Begin illumination /* Store illumination level before light source contribution */
    surface:my_lighting_channel = copy(surface:illumination)
Surface illumination
    .... /* Lighting actions here */
End illumination /* Compute how much illumination increased */
    surface:my_lighting_channel = subtract(surface:illumination, surface:my_lighting_ch
```

Initialized channels:

All Surface sample channels that were initialized in the Surface properties shader and modified by Surface illumination shaders.	

Surface Finishing

This shader is the very last step of surface shading. In this shader, one can perform actions that control the total surface shading. For example, one can compute all illumination contributions to a buffer channel and then define the actual illumination using this shader.

Initialized channels:

All Surface sample channels that were initialized in other surface shaders	This is the last surface shading step, so most channel values have already been computed.

Channels used by the rendering system after the shader:

Illumination	Illumination channel contains the result of ray tracing.

Volume Illumination

This is similar to the surface illumination, except that this time, illumination of an interior point of a volume is examined. By default, ray tracer does the diffuse shading effect by the given color channel value: light makes the fog glow if the color channel is not black. The diffuse shading depends only on the intensity of the light in the examined point, it does not depend on the lighting direction.

Initialized channels:

All Light sample channels that were initialized in the Light properties shader	See 'Light properties' above.
All Volume sample channels that were initialized in the Volume properties shader	See 'Volume properties' above.

Channels used by the rendering system after the shader:

Light:Illumination	The amount of light.
Volume:Color	Diffuse lighting depends on this.
Volume:Illumination	The result of volume illumination. Add custom illumination to it.

Volume Shading

This shader is executed as the last step of volume processing. At this point, the Volume sample contains the fully illuminated fog in the 'Volume:Illumination' channel. The remaining step is the default shading, which blends the fog illumination with the result of ray tracing using the distance and turbidity. The thicker the turbidity and the longer the distance, the more fog illumination affects the result. VSL code in 'Volume shading' can make the final adjustments before the default shading. It can also compute completely customized fog shading and then set the 'Turbidity' to zero to eliminate default shading.

Initialized channels:

Volume:Distance	The distance between samples (=diameter of space area the sample represents).
Volume:Illumination	The result of fog effects.
Volume:Turbidity	The density of fog.
Ray:Distance	The total distance the ray travels in the volume.
Ray:Illumination	The result of ray tracing before fog effects.

Channels used by the rendering system after the shader:

Ray:Illumination	The result of ray tracing before fog effects. You can compute a custom effect and blend it to this channel.
Volume:Turbidity	The density of fog. Defines the strength of fog effect. Set to zero to eliminate the default fog shading.
Volume:Illumination	The result of fog effects.

Initialize Ray

This shader is called when tracing of a new ray (reflection, GI ray etc.) starts. It is not called for rays originating from the camera. In this shader, you can move data from the originating surface into the traced ray.

Initialized channels:

Traced ray:Coordinates, Traced ray:Ray	The origin and unit direction of the traced ray.

Surface sample channels initialized by the surface properties and surface illumination shaders	



Note

If the ray is generated by a user defined raytrace object, the set of initialized channels depends on the location of the raytrace object in the VSL shader tree.

Channels used by the rendering system after the shader:

Traced ray:Coordinates, Traced ray:Ray	The origin and direction of the ray to be traced.

Secondary Ray

This shader is executed when tracing of a ray is finished and the remaining step left is to add the new information to the overall result of shading. For example, fading of reflection by distance can be done in this shader.

Initialized channels:

All Traced ray channels that were initialized by other shaders encountered while tracing the ray.	See the shaders above.
Surface sample channels initialized by surface properties and surface illumination shaders.	See 'Surface properties' and 'Surface illumination' above.

Channels used by the rendering system after the shader:

Surface:Illumination	The result of rendering so far.
Traced ray:Illumination	The result of 'child' ray tracing.

The default shading operation, which combines the two generations of rays, is simply
 $\text{Surface:Illumination} += \text{Traced ray:Illumination} * \text{Signalcolor}$

Here 'Signalcolor' is the channel that created the new ray, such as transparency or reflection.

Ray Hit

This shader is quite a special one. The ray tracer executes it when computing distances to a surface. By modifying the distance, one can distort the surface. If the distance is set to a negative value, the surface disappears and so-called clip mapping effect is achieved.

Initialized channels:

Coordinates	The hit position.
Distance	The distance from the ray source to the hit position.
Map coords	Mapping coordinates.
UV coords	Surface coordinates.

Channels used by the rendering system after the shader:

Distance	Defines visibility of the surface.

Ray Termination

This shader can be used for some fine detail effects for transparent objects. The ray tracer executes it when the tracing of a ray is stopped because of finite recursion depth. Usually, one could add the average of the color of the overall scene to the illumination channel.

Initialized channels:

All surface sample channels that were initialized by the Surface properties and surface illumination shaders.	See 'Surface properties' and 'Surface illumination' above.

Channels used by the rendering system after the shader:

Illumination	Defines the result of tracing the current ray.

Scanline

This shader allows rendertime displacement of scanline objects using materials. The density of points which are passed for this shader depend on the particle type. For example, a particle curve evaluates point displacement densely along the curve, and this shader can be applied to make it curly. Particles may also support other control channels (diameter channel etc.)

Initialized channels:

Coordinates	The position of a geometry point on a particle.
Map coords	Mapping coordinates for displacement parametrization.
UV coords	UV coords run along the surface of each particle, describing its geometry.

Channels used by the rendering system after the shader:

Coordinates	Change this to achieve displacement. Note Use 'Transform to material space' option when computing the displacement vector. Rendering space relative displacement will not behave consistently in animations.

Post Particles

This shader is executed when rendering particles in the post-processing module. For example, it is possible to define the color of particles in this shader. The purpose of this shader is similar to that of the 'surface properties' shader, but as the particle rendering in post processing is a much simpler operation than surface rendering in the ray tracer, less channels are available.

Control channels:

Color	The default diffuse color (white) or particle's color.

Fade	Controls transparency for most effects. Fade 1.0 makes particles totally invisible.
Coordinates	The position of the particle, all particles compute this. Change it to displace the particle.
Map coords, Map x, Map y, Map z	Mapping coordinates and orientation.
U direction, V direction, Normal	Additional information of particle geometry. See the particle specific documentation for details.
Time	The global time.

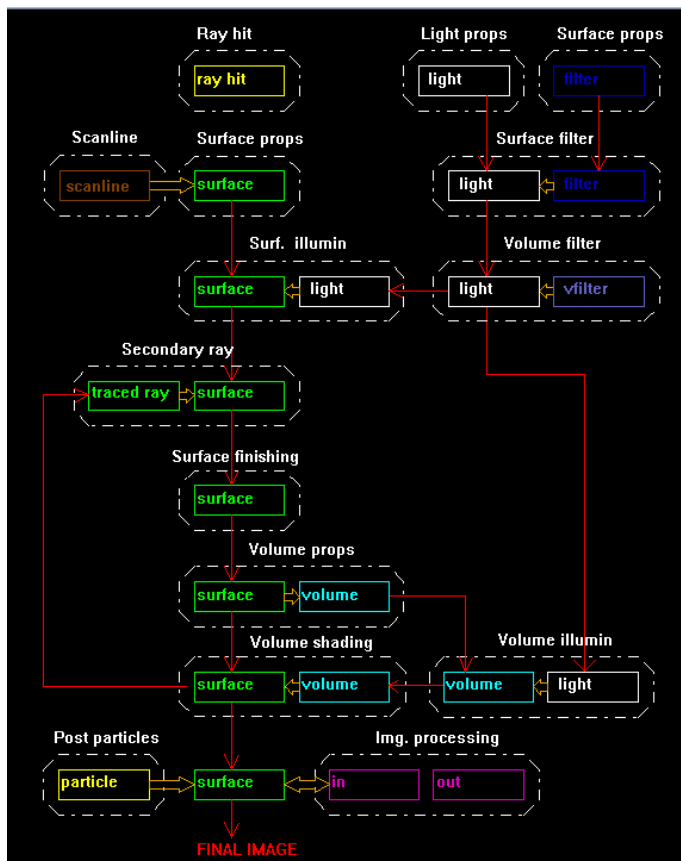
Image Processing

This special shader is plugged in by one post processing effect, 'material effect'. The purpose is to let the user manipulate pixels of the rendered image by a shader system. As described before, the material editor of the program includes a powerful programming language, and this shader allows the user to define custom image processing effects with the language. All the channels are available in this shader.

Input channels	All channels.
Output channels	All channels.

Rendering Diagram

The picture below shows how the rendering system moves data between the samples. Thick orange arrows represent internal default shading operations between samples, filling the gaps in the data flow.



VSL Objects

VSL objects compute and assign values to channels. A VSL object can be a mathematical formula, transformation curve or an algorithm, to name a few examples.

There are hierarchical, conditional and procedural objects available, as well as local variables. Also recursive structures are possible. The included object set provides the user with a visual programming language, which is specialized in creating algorithms for shading effects. There are tens of channels available, and an extensive set of objects with a variety of options; the number of possible combinations created using these elements is unlimited.

Most built-in and also user defined custom objects, if implemented properly, are 'reusable'. For example, a texture mapping object can be assigned to the transparency channel as well as to the color channel. Nevertheless, some objects have quite a special purpose: for example, the computations made by the bump map object are specifically designed for modifying surface orientation. One can assign the bump map object to the color channel, but the result hardly makes sense.

Bump

A Bump object uses a bitmap image to define a 'height transformation' to its output channel. This transformation is computed in such a way that, when assigned to the 'Bump normal' channel, the surface appears bumpy. The controls of a bump map are:

- *File name*: The name of the image file used for bump mapping.
- *Tile X* and *Tile Y*: These controls repeat the bump map pattern on the surface of the target object.
- *Flip X* and *Flip Y*: These options work in combination with the tile options. If Flip X is set, every second tiled bump area is inverted horizontally, so that the left and the right edges of adjacent tiles always match. Flip Y does the same in the vertical mapping direction.
- *Bleed X* and *Bleed Y*: Bleed option extrapolates edge values outside bump mapped area.
- *Interpolation*: Interpolation type defines the shape of the bumps. Linear interpolation produces sharp ridges and peaks, whereas Cubic produces very smoothly varying bumps. Bi-linear and Quadric are between these two.
- *Compute Direction/Height*: Compute Direction is the suitable operation mode when a bump map is used in a Surface properties shader for direct bump normal manipulation. Compute Height is designed for use in a Surface geometry shader.
- *Bump height*: Defines the height of bumps. In Compute Height mode, the measuring unit is meters, and therefore quite small values are often sufficient.
- *Smooth*: A smoothing filter. This feature eliminates pixelization of high resolution bump maps. The measuring units is the proportion

radius of the smoothing filter/image width

Therefore, quite small values are usually sufficient. Level 0.01 smoothing in a 500*500 pixel image uses a $10 \cdot 10 = 100$ pixel filter for smoothing.

- *Binding*: Binding controls are included in a separate tab when a bump object is selected. The tab displays a list of data channels, which are included in the selected image. By default, the first data sub channel (red in a usual color image) defines the bump height, but other channels can be selected as well.

Animation tab: This tab includes controls needed when using an animated image sequence for defining the bump map (or any other texture map type). This means that the program automatically loads a new bump map in every rendered frame.

- *Animation Type*: This option defines whether the bump map animation is on or off, and how the image index should be computed. The following alternatives are available:
 - *None*: Animation disabled. The first frame of an animation file or the specified image without index changes is used.
 - *Automatic*: The image index is computed using 'Repeat', 'Base index' etc. options automatically. By default, the full image cycle is used repeatedly.
 - *Key Frame*: The image index is computed from user defined key frames. Index does not change before a key frame sequence is recorded using the 'Current Index' field.
- *Current Index*: Shows the texture index which will be used in the current animation frame. If 'Key Frame' Animation type is selected, you can enter indices to this field in animation recording mode and hence key frame an arbitrary order for texture indexing.
- *Base index*: Defines, what is the index of the first image of the bump map sequence (usually 0 or 1). The value can also select the start of a sub range of a bump map sequence.
- *Frames*: Number of map images used for the animation (usually the total number of images in the map sequence). The images are selected starting from the base index. For example, consider a ten image sequence named as 'map0', 'map1', ..., 'map9'. If 'Base Index' is 3 and 'Frames' is 5, the index sequence will be 3, 4, 5, 6, 7, 3, 4, ...
- *Repeat*: Activates repeated cycle of bump map images. First image is loaded again after the last defined image.
- *Ping Pong*: Activates 'ping-pong' style repeating of bump map sequence: bump index increases until the last index is reached, then it starts decreasing until the first index is reached, and so on.
- *Reverse*: Inverts the direction of bump map sequence.
- *Format Type*: Defines how names of separate images are indexed (a sample name is displayed below). When 'Default' indexing is set, the number of digits can be defined with the 'Digits' slider. When 'Custom' indexing is selected, any kind of indexing convention can be defined using the 'Format string' field.
- *Digits*: Number of decimal digits for image index. The value controls zero padding for small indices.
- *Format String*: Custom indexing format. The string follows the standard C language 'printf' conventions.



Note

Automatic texture indexing changes the index in every frame, field and motion sample, so that texture animations are motion blurred as well. Other kind of behavior can be defined by animating the texture index by key framing. The choreography window makes it possible to modify the animated indexing in a highly flexible manner: the index curve can be point edited, the time line can be moved in time, scaled, repeated, etc.

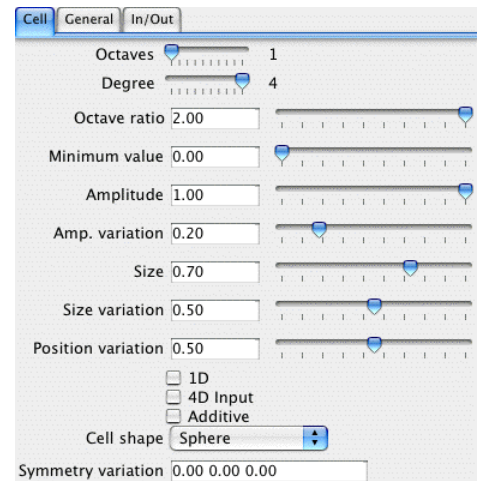
Call

This object can be used to execute a user-defined VSL procedure (see also VSL Procedure Library). The Attribute tab of the Call object displays a list of available user-defined procedures for selection. After selecting the call target, the user can set the respective input and output parameters.

Cell

Cell noise is a versatile VSL object, which is especially useful in bump mapping. Cell noise is a combination of 'noise bubbles' in 3D space. It has the following options:

- **Octaves:** The amount of iterations in cell noise computation. The final cell noise is a sum of several iterations of cell computation. Every iteration has 'Octave ratio' times higher frequency and lower amplitude.
- **Base value:** Defines the base value of the noise signal.
- **Amplitude:** Defines the maximum range from the base value.
- **Amp. Variation:** The amount of random variation in amplitude between cells. 0 = All cells have equal amplitude, 1=Amplitude varies from zero to maximum amplitude.
- **Size:** The relative size of cells. The size 0.5 produces cells that do not overlap but slightly touch each other (if variations are zero).
- **Size variation:** The amount of variation in cell size. 0=all cells have equal size, 1=size varies between zero and the maximum value defined by Size.
- **Position variation:** The amount of random variation in cell positions. 0=all cells are in a regular 3D grid, 1=two adjacent cells may overlap completely.
- **1D:** The cell noise is one dimensional. 1D noise is faster to compute than 3D cell noise.
- **4D Input:** Selects four dimensional input for the noise. When the option is set, the cell object gets a second input channel, whose first sub channel is used for the additional 4th dimension. Typically the 4th dimension is animation time. This is useful for creating animated turbulent fog, fire and other similar phenomena.
- **Additive:** If set, the noise is sum of adjacent cells. Otherwise, the maximum value of adjacent cells defines the noise.
- **Centered:** If set, Amplitude varies symmetrically around the Base value. If cleared, Amplitude is added to the Base value.
- **Cell shape:** Sphere - spherical cells, Cube - box like cells, Rounded cube - Cell shape is a box with rounded edges.
- **Asymmetry:** Defines how symmetric the cells are. If cell shape is sphere and asymmetry is zero, all cells are spheres. Non-zero asymmetry produces elliptic cells. The value 1 makes the proportions of cells fully randomized. The value of this attribute is a vector, and therefore variation of each cell dimension width/height/depth can be defined independently.
- **Density:** Scales the input parameter. The higher the density, the smaller (=denser) patterns the cell noise creates.



Comment

The purpose of this simple object is to add some optional information into the VSL shading tree. It does not perform any operation during rendering.

The comment string can be changed by a slow double click over the name string or by applying the Rename tool of the popup menu of the VSL tree gadget.

Constant

A Constant object assigns a constant value to its output channel. The type of its value is automatically chosen according to the output channel type.

Copy

This is a 'data mover' object. Copy object copies data from its input channel to the output channel. It is possible to choose which input sub channel is copied to which output sub channel. It is also possible to leave some output sub channels intact.

Curve

A Curve object allows the user to define an arbitrary transformation, defined by easily editable curves, between the input and output channels.

The number of displayed curves depends on the dimension of the output channel. A float output channel can be transformed by only one curve, whereas a color or a vector channel requires three curves, one for each sub channel.

Curves can be edited easily. The object automatically inserts a new point to the curve, if user clicks a curve in such a location where there is not yet a point. Existing points can be selected by clicking and multi-selected by shift-clicking or dragging. Points can be deleted using the popup menu of the curve gadget. For more information about curve editing, check the curve editor documentation.

There are five curve types available:

- *Linearly interpolated*. Points are connected using straight lines.
- *Boolean*. The y value of the curve changes only at control points.
- *Nurbs*. The standard cubic Nurbs curve.
- *PNurbs*. This interpolation method produces a stable curve also from badly oscillating data. Nevertheless, if interpolation data is bad, continuity will not be quite as good as when using regular Nurbs interpolation which is always C2-continuous.
- *Bezier*. The standard cubic Bezier curve. This curve type can represent both smooth sections and sharp angles.

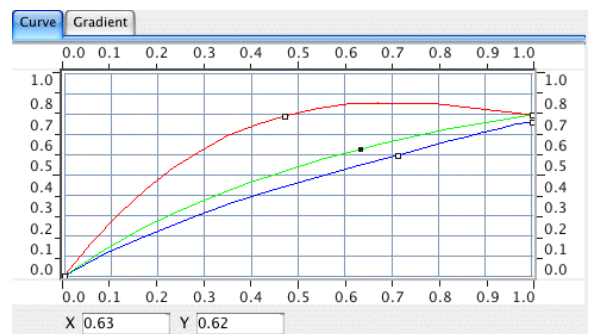
The input and output range of the curve transformation can be adjusted using the menu 'Set minimum and maximum values' from the popup menu. The default range is 0-1. The value of the transformation outside the defined input range is defined by the values at the ends of the curve.

The Gradient option defines another kind of interface for curve editing: the result vector of a transformation is displayed as an RGB color. This is useful when designing color gradients and other RGB transformations.

Derive

Mathematically speaking, the object computes u and v partial derivatives of the norm of the output channel and assigns the computed two values to the output channel. Then it transforms the result to the surface orientation and normalizes the result. Practically speaking, these are exactly the steps that transform any function to a bump-map suitable form.

Usually, at least some of the objects use the same input channel (i.e. the channel with respect which the derivative is computed) as their parent 'Derive' object. Also, at least one of the sub objects should assign a value to the same output channel the derive object has (otherwise the derivative would be null).



Editing a curve object. The red curve maps the first sub channel value 0.479 to a new value 0.643.

The Derive object is mainly included for backward compatibility reasons. It has its use when computing a bump normal modification in the 'Surface properties' shader, but the same effect can be achieved in a simpler manner using a 'Surface geometry' shader.

Else

Else object defines a VSL level, whose sub objects become evaluated when conditions of a preceeding If object were not set. An Else object without an If object placed before it at the same hierarchy level does nothing (its children are never evaluated).

Else object does not have to be immediately after the If object: other VSL objects can be evaluated between the two.

If

This is the conditional VSL object. It is a 'level' type of object: other VSL code can be placed inside it. The object defines a condition, and sub objects are executed only when the condition is met.

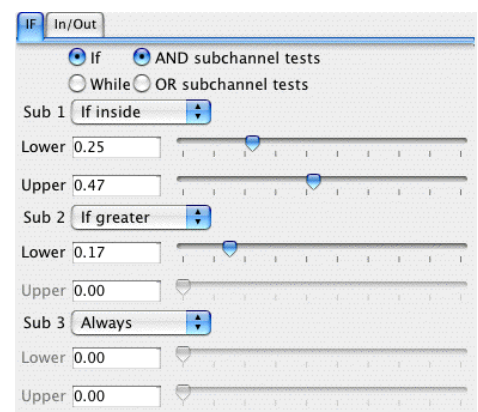
If object can be paired with an Else object, that follows the If object in VSL hierarchy. Its contents are evaluated when If condition are not met.

If-object can be used in two ways: for single execution of sub objects ('If' option selected) or for continuous execution ('While' option selected). The condition is defined for each sub channel separately. It has five alternatives:

- *Always true*: the sub channel does not set any condition
- *If inside*: true if sub channel value is inside a given range
- *If outside*: true if value is outside a given range
- *If greater*: true if value is greater than or equal to the given value
- *If less*: true if value is less than or equal to the given value

The channel sub tests are combined using a suitable logical operation:

- *AND operation*: The sub objects are executed if all sub channel conditions are true.
- *OR operation*: The sub objects are executed if at least one sub channel condition is true.



Level

The level object can be used in collecting VSL objects into hierarchical groups. A VSL level also defines a scope for variable objects inserted into it. A level object has no options.

Linear

This object computes a usual 'linear' transformation: it multiplies the input by a given factor and adds a constant value to it.

Map

Map object computes the mapping coordinate projection from the given parameter. The object is compatible with all material mapping objects which use a 3D geometry to define the mapping function. Surface defined coordinates (vertex variable interpolation etc.) are not supported.

One important use of the Map object is to provide a consistent parametrization for randomized sampling. For example, a volumetric fog shader may change the sampling position slightly to examine the surrounding volume better:

```
volume properties
  vector variable pos
  vector variable mc
  vector variable delta
  delta = random(map coords)
  pos = add(coordinates, delta)
  mc = Map(pos)
  turbidity = Noise(mc)
  ...
```

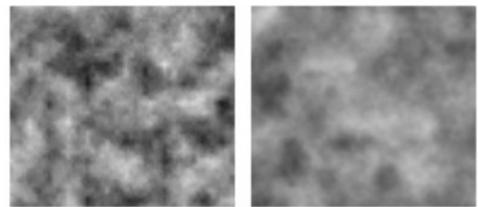
This kind of fog density definition appears consistent for a moving camera, self-shadowing fog etc.

Noise

Noise is a very useful object, which can be used in simulating various 'fractal' phenomena.

It has the following controls:

- *Octaves* defines the amount of iterations used in computing the fractal noise. The higher the value, the more detail the noise has (but it may take slightly more time to compute it).
- *Octave ratio* field defines the fractal dimension value: $\text{total noise} = \text{noise}(\text{input}) + \text{noise}(\text{input} \times \text{dimension}) / \text{dimension} + \dots$
- *Base value* defines a base for the noise signal.
- *Amplitude* defines how much the noise values range from the base value.
- *Density*: Scales the input parameter. The higher the density, the smaller (=denser) patterns the noise creates.
- *Smooth* option computes smoothly continuous noise. This makes computation slightly slower, but the result is more suitable, for example, for bump mapping.
- *1D noise* computes one dimensional output values. If the output channel is a vector, all sub channels get the same value. 1D noise is faster to compute than 3D noise.
- *4D Input* selects four dimensional input for the noise. When the option is set, the noise object gets a second input channel, whose first sub channel is used for the additional 4th dimension. Typically the 4th dimension is animation time. This is useful for creating animated turbulent fog, fire and other similar phenomena.



Linear and Smooth 1D Noise assigned to the Color channel

- *Centered* If set, Amplitude varies symmetrically around the Base value. If cleared, Amplitude is added to the Base value.

Operation

This object offers a wide collection of mathematical formulas and functions, including the following (the first input is denoted as p1, the second as p2, etc.):

- abs. value - absolute value, inverts sign of negative values
- add - addition of two channels
- arc cos - usual trigonometric inverse functions
- arc cot
- arc sin
- arc tan
- ceiling - rounding up to integer value
- cos
- cotangent
- cross product - computes cross product of inputs, the result is perpendicular to both
- divide - $p1/p2$
- dot product - $(x1, y1, z1)$ dot product $(x2, y2, z2) = (x1*x2, y1*y2, z1*z2)$
- exponent - $\exp(p1)$
- faceforward - computes a vector which is parallel to p1 and faces against p2. In other words: $v = \text{faceforward}(\text{unit\_normal}, \text{ray})$ means that $v \cdot \text{unit\_normal} = +1$ and $v \cdot \text{ray}$ is less than 0.
- floor - rounding down to integer value
- invert - $1/p1$
- length - the length of the input vector = $\text{square\_root}(x*x + y*y + z*z)$
- logarithm - natural logarithm
- max - $\text{maximum}(p1, p2)$
- min - $\text{output} = \text{minimum}(p1, p2)$
- modulo - $p1\%p2$ (floating point modulo)
- modulo by 1 - $p1\%1$, the decimal part of p1
- multiply - $p1*p2$
- negate - $-p1$

- normalize - $(x, y, z) - (x, y, z)/\sqrt{x^2 + y^2 + z^2}$
- polar - transforms cartesian coordinates to polar coordinates (heading, pitch, distance)
- power - p1 to the power of p2
- reflect - reflects a ray p1 with respect to a unit normal vector p2.
- refract - computes light refraction refracted ray=refract(exterior thickness, interior thickness, incoming ray, surface normal)
- round - rounding to the nearest integer
- sign - -1 if parameter is negative, 0 if zero, +1 if positive.
- sin
- sqrt - square root
- subtract - p1-p2
- tangent
- $1-(1-p1)(1-p2)$ - some useful formulas
- $(x,y,z)-xyz$
- $1-p1$
- $(1-p1)*p2+p1*p3$ - linear interpolation
- $(p1+p2)/2$ - average of input channels
- $(x,y,z)-(x+y+z)/3$ - sub channel average. Can be used in color to gray scale conversion, etc.

Procedure

This VSL object defines a general purpose VSL function which can be called from other VSL code. Procedures can be created only into VSL Procedure Library objects. A procedure has the following properties:

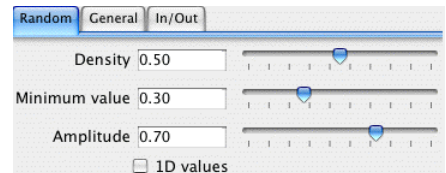
- **Name** - A name for the procedure. You can set it to describe the purpose of the included VLS code. For example "Gamma Correction", "SuperNoise" etc.
- **Parameters** - Procedures cannot refer to any external data, all channel information must be passed in and out using parameters. The Parameter tab of the VSL editor has the following controls:
 - *Parameter List* - The complete list of procedure parameters.
 - *Add* - Adds a new parameter to the procedure.
 - *Delete* - Removes the selected parameter. Note: If there is already VSL code calling the procedure, their Call parameters will not be automatically remapped.
 - *Parameter Name* - It is a good idea to give descriptive names to parameters, especially if the procedure has many of them. You can then easily recognize the parameters later when defining calls to the procedure.
 - *Variable Type* - The data type of the selected parameter. You can guide the program to use appropriate gadgets for parameters by setting the type. For example, if the procedure defines a color transformation, you may set

its input parameter type to Color. Then for example Constant and Linear VSL objects show you a color gadget when you assign a value to the parameter.

Random

This VSL object computes random values. The following controls are provided:

- *Density* defines a probability threshold for the random computation. If the density is 0.1, only 10 % of evaluations produce a value other than the random base value. If the density is 1, all evaluations of the object compute a random value.
- *Base value* defines the value, which is output if the density threshold is not triggered. It also defines the base level for random values.
- *Amplitude* defines how much values range from the base value. If the base value is 2 and range is 1, the random values vary between 2 and 3 (when Centered is cleared).
- *1D values* option selects 'gray scale' output: all output sub channels get the same random value.
- *Centered* If set, Amplitude varies symmetrically around the Base value. If cleared, Amplitude is added to the Base value.



Raytrace

Raytrace object gives an access to the actual ray trace engine of Realsoft 3D. It is possible to create customized ray tracing effect using this object.

Some examples:

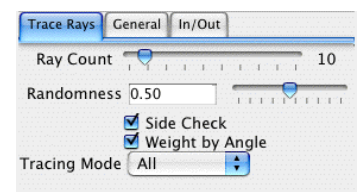
- Blurred reflections by sampling 5-10 rays around the reflected ray
- Global illumination by sampling some random rays around the surface normal
- Even fog can reflect the surrounding illumination. Each fog volume sample can trace some rays to examine what is around the position.

Raytrace object requires 2-4 input channels:

- input1: ray origin (usually 'Coordinates' channel)
- input2: ray direction (e.g. 'Reflected ray' or 'Normal'). Does not have to be a unit vector.
- input3: weight color (signal strength of the traced rays). The illumination value of the traced ray is multiplied with this value.
- input4: an optional half space selector for keeping randomized rays in the appropriate side of the defined direction. The 'Surface:Normal' channel is usually suitable for this purpose. Input4 is also used when 'Weight by Angle' is set, in which case the value should have unit length.

The raytrace object outputs the result of ray tracing which is usually added to the contents of the illumination channel.

The object has the following options:



- *Ray Count*: defines how many rays are traced.
- *Randomness*: defines how much the traced rays are scattered randomly around the defined main ray direction. 0=no randomness, 1=fully randomized ray direction (up to 180 degrees from given ray direction)
- *Side Check*: if set, the ray randomization is done so that the rays remain on a given half-space (e.g. on the same side of a surface). The side is defined by the parameter input2 (ray direction) and input4 (side selector)
- *Weight by Angle*: If set, the contribution of each ray is weighted by the dot product `sampled_ray*input4`. This is a useful option for illumination effects, input4 being the surface normal.
- *Tracing Mode*:
 - *All*: Performs full recursive ray tracing and stores the computed illumination to the output channel.
 - *Distance*: Finds the only the nearest ray hit and stores the respective distance to the output channel.
 - *Occlusion*: Stores the ratio `hit_rays/all_rays` to the output channel. The value can be used by Ambient Occlusion like illumination shaders.

**Note**

Occlusion and distance rays do not interfere with other shaders, which process or manipulate traced rays.

**Note**

Trace Range channel controls all tracing Modes of the Raytrace object.

Shader

This object has the special task of dividing the execution of the VSL material into parts. Material properties are needed in various situations, and evaluating all properties every time would be slow. Therefore, the highest hierarchy level of a VSL material consists of shaders, which are levels under which the actual VSL code is placed. The type of shader defines when the included code is executed. More information can be found in the chapter 'Shaders'.

Shadow Map

Shadow map object is the same object which the render engine uses for mapped shadows. Render engine generates the required maps automatically as a part of the rendering process. With this object, you can do the same 'manually'. For example:

- Create a camera that matches the beam of a spot light
- Render an image file (preferably using Realsoft 3D's native multi channel format) that contains transparency and distance channels
- Map the image to a light source using the Shadow Map shader and a pyramid mapping which matches the camera geometry

Shadow Map has the following controls:

- *File* defines the name of the shadow map file.
- *Smooth* defines how much smoothing is applied to the shadow map. The value defines the proportion smoothing width/image width, so quite small values (0.01 - 0.1) usually suffice.

- *Bindings/Transparency channel* selects the image channel from which shadow transparency is read.
- *Bindings/Distance channel* selects the image channel which defines nearest shadow distance for each map pixel.

Specular

This object computes a specular illumination effect. It has two controls:

- *Sharpness* defines the size and sharpness of specular illumination effect
- *Brightness* defines the brightness of specular illumination effect



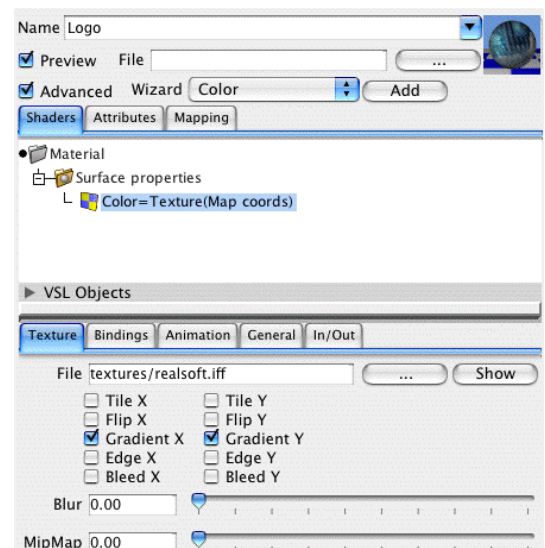
Note

Operation += is automatically selected when a specular shader is dropped to the VSL hierarchy. This is because of the additive nature of specular illumination.

Texture

This is the common texture map object. Its controls are:

- *File name*: defines the name of image file.
- *Tile X* and *Tile Y*: these controls repeat the texture map on the surface of the target object.
- *Flip X* and *Flip Y*: If Flip X is set, every second tile is inverted horizontally, so that the left and the right edges of adjacent tiles always match. Flip Y does the same in the vertical mapping direction. The option can also be used without tiling to mirror texture maps.
- *Gradient X* and *Gradient Y*: Smooth interpolation between pixels. Boundaries of pixels of texture image become blurred, which in many cases improves quality.
- *Edge X* and *Edge Y*: By default, gradient interpolation is extended over the edge of the texture map to the background color. Edge X and Y eliminate gradient on the edges.



Texture mapping in action

- *Bleed X* and *Bleed Y*: Bleed option extrapolates texture edge colors outside texture mapped area.
- *Blur*: A smoothing filter. The measuring units is the proportion

radius of the smoothing filter/image width

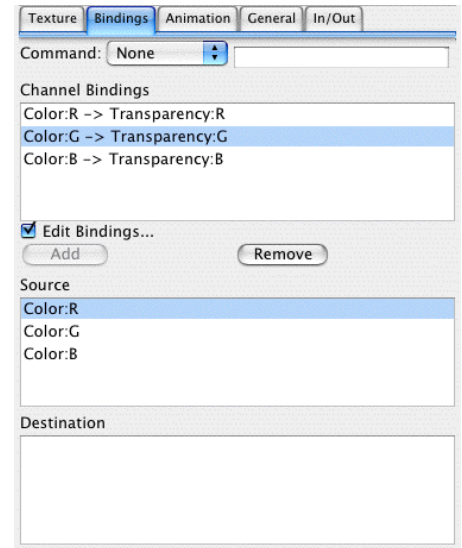
Therefore, quite small values are usually sufficient. Level 0.01 smoothing corresponds one percent of the image width which usually blurs the image quite a lot. Note, that 'Gradient' feature also smoothens the pixel boundaries, but smoothing uses only 2*2 pixel blocks. Blur smoothing is not limited, it can use e.g. 10*10 pixel area or more for blurring.

- *MipMap*: A special antialiasing technique for texture maps. If the value is greater than zero, the program computes a sequence of mipmap images. Each image halves the resolution of the previous one. The last image is only one pixel wide and contains the average color of the whole original image. The render engine uses interpolated colors from mipmaps whose resolution is suitable for the image area being computed. The higher the MipMap value, the more low resolution mipmaps contribute into the result.

The Bindings tab of the texture object gadget includes tools for specifying how to use the image data. It displays so called channel bindings, and allows the user to redefine them.

The Edit Bindings part of the tab shows the available image channels and the sub channels of the output channel. The existing channel bindings can be selected and then removed using the Remove button. New bindings can be created by selecting an image channel and an output sub channel and by pressing the Add button.

The Animation tab of the Texture object is the same as the Bump object has. See Bump documentation above for details.



Editing sub channel bindings

Variable

The 'Variable' object defines a 'local VSL channel'. A variable is visible everywhere within its parent level and in all the parent's sub levels. Variables can be used to store temporary results.

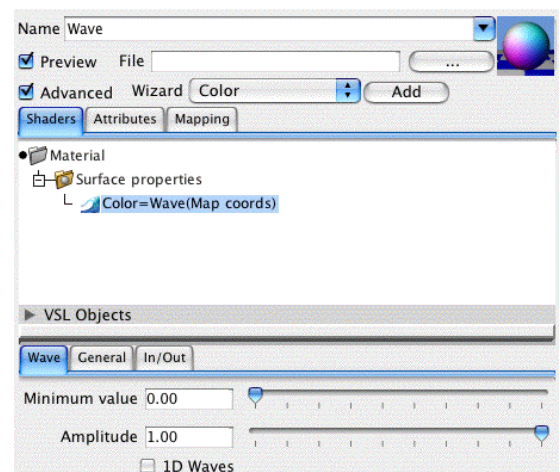
Variables have two attributes: name and type (float/vector/color).

The root level variables are different from other variables in that they are 'static'. This means that the storage for them is allocated once in the beginning of rendering and it does not change during successive evaluations of the material. This means that root level variables can be used to store constant data, which is initialized only once at the beginning of rendering (in a 'Material initialization' shader). Actually this also means that root level variables can pass data between shaders, but this capability should be used with care, because it may be difficult to ensure that the material is not evaluated again between writing to and reading from the variable. Ray tracing is a recursive process and the same material may be encountered again when following a reflected ray, etc. Therefore, user channels are a better alternative for this kind of purposes.

Wave

This object computes a trigonometric sine wave function. Its controls are:

- *Base value* of the wave. The output value on the bottom of the wave.
- *Amplitude* defines the height of the waves.
- *1D values* option selects 'gray scale' output: all output sub channels get the same wave height value, computed from the first input sub channel.
- *Centered* If set, Amplitude varies symmetrically around the Base value. If cleared, Amplitude is added to the Base value.



Color waves

VSL Wizards

The easiest way to create new kinds of VSL materials is to use VSL wizards. These wizards are available in the VSL editor. Wizards construct the necessary material structure automatically, and insert suitable controls for easy use.

Bump map	Creates a standard bump map that makes a surface appear bumpy. The controls available are: • Bump Height • Tile X: Horizontal tiling • Tile Y: Vertical tiling • Interpolation: Type of bumps. Linear=sharpest bump shapes, Cubic=smoothest bump shapes. • File name: The name of the bump map file
Chrome	The Chrome wizard creates a mirror like, shiny material. The only control available is the color of the material.
Clip map	This wizard creates the clip map effect. It is possible to define the color range, which removes the underlying surface. Inserted controls: • File name: The name of the clip map file • Sub channel 1 min & max: These two values define the clipped red signal range • Sub channel 2 min & max: These two values define the clipped green signal range • Sub channel 3 min & max: These two values define the clipped blue signal range Note that all the conditions must be satisfied before the clipping happens (red inside the red clip range AND green inside the green clip range AND blue inside the blue clip range)
Color	This wizard adds an object that controls the usual diffuse color. A new color value control is inserted for setting the color.
Density	This wizard controls material's mapping parameter density. Texture patterns, bump maps etc. effects can be scaled using it.
Fog	This wizard creates a fog material that has transparent surface and constant density.

	The fog has 'plasma' behavior, in other words, its illumination does not depend on lighting conditions but is self illuminated. The fog attributes are: • Fog color: Defines the color of fog. • Fog density: Defines the density of fog. The higher the density, the more visible fog appears, and the more it hides objects inside and behind it.
Glass	The Glass wizard creates quite realistic glass. The surface of the material has a varying amount of subtle reflection, and the material also has a semi-transparent, 'caustic' shadow. Controls are: • Glass color: The color of the glass • Refraction: The amount of refraction. 0=no refraction, 0.4=standard glass, 0.9=very strong refraction.
Glow	Glow wizard adds two controls for defining glow: brightness and size. Glow brightness should usually match object color, and glow size, being measured as meters, is typically quite small (default 0.05). The glow effect must be included in the post effect configuration when using glow materials.
Reflection	This wizard adds an object that controls mirror like reflectivity of the material. The reflection color can be set using the inserted color gadget. Black reflection color removes all reflections, whereas white reflection color produces chrome like, clearly visible reflections.
Self Illumination	This wizard adds a given RGB amount of self-illumination (object-specific ambient lighting) to the material.
SpecularColor	Inserts VSL objects for specular illumination effect. The controls are: • Sharpness: Defines how small and sharp-edged the specular illumination effect appears. • Brightness: Defines the maximum brightness of specular illumination.
Steel	This wizard simulates satellite metal, which is only partially reflecting and whose surface is slightly rough. The included controls are:

	• Metal color: Color of the material. grayish blue=steel, dark orange=copper, etc. • Roughness: Defines how smooth the surface is. • Sharpness: Sharpness of specular lighting effect • Brightness: Brightness of specular lighting
Texture map	The Texture wizard creates a standard texture map object that defines object's color. The controls available are: • Tile X: Horizontal tiling • Tile Y: Vertical tiling • Grade X: Horizontal smoothing • Grade Y: Vertical smoothing • File name: The name of the texture file
Transparency	This wizard adds an object that controls the transparency of the material. The level of transparency can be defined using the inserted color gadget. Black transparency color means full non-transparency, white transparency produces glass-like transparency. Note that usually the color wizard should be used in combination of this wizard, because glass-like objects have (nearly or totally) black diffuse color.

Output Objects

Output Objects define a rendering target, such as a file format. Such objects are needed in file rendering. The program includes various output object types, which have different properties: JPG allows good compression, AVI objects can store several consecutive frames, Targa supports alpha channel etc. The property window controls described below reflect these differences.

The common properties of all output objects are:

Name: A symbolic name for the object. It usually reflects some characteristic properties and the file format of the object: Jpeg good quality, Preview window, Targa RGBA etc.

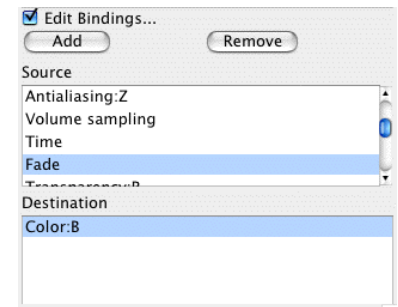
Command: A script language or OS command line string that will be executed after the image data output to the object has been finished. The string can include any of the following tokens (square brackets must be included), which the program replaces with the actual used names before executing the command:

- *[file]* - The total file name, including path, possible frame index and extension.
- *[path]* - The path to the output file.
- *[base]* - The output file name without the path and the extension.

- *[ext]* - The file format extension without the dot separator.

The output object command feature can be used for automatic file format conversion using a command line based image processing program such as Image Alchemy.

Channel Bindings: Bindings define what rendering data is placed into the channels of an output object. Each output object initializes reasonable binding defaults at the object creation. For example, the red, green and blue components of the total rendered illumination of each pixel go to the corresponding components of the usual color channel available in all output formats. This default behavior can be customized with the binding editor, which has the following controls:



- *Channel Bindings* - This list shows the current set of bindings. Items at the left side of each line are Realsoft 3D's data channels defined by the rendering modules and shaders. At the right side of the arrow is the sub channel of the image format to which the data is written.
- *Edit Bindings...* - Displays the binding editor where channel bindings can be changed.
- *Add* - After selecting an unused image format sub channel and a Realsoft 3D rendering data sub channel, you can press 'Add' to create the new channel binding.
- *Remove* - To remove an existing binding, activate an item from the 'Channel Bindings' list and click this button.
- *Source* - The list of all rendering data sub channels of Realsoft 3D.
- *Destination* - The list of all (not yet bind) image sub channels. When you bind an image sub channel, it disappears from this list. Deleting a binding brings the sub channel back to this list, so that it can be bound to another rendering data sub channel.

AVI

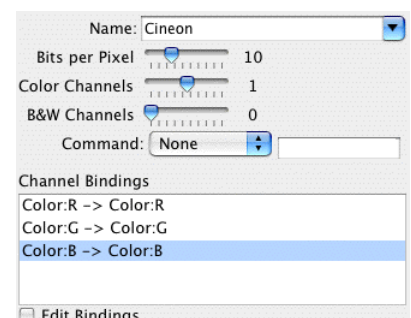
The AVI specific options are:

- **fps**: The preferred playback speed as frames per second.
- **Video Compression**: Opens a dialog where the preferred compression method and its settings can be selected. The compression types and settings available depend on the operating system configuration and other installed software packages.

Cineon

The Cineon format has the following options:

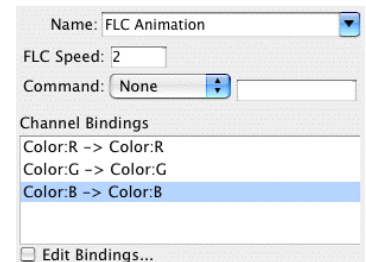
- **Bits per pixel**: The data bandwidth as bits per sub channel.
- **Color Channels**: The amount of RGB channels (1 or 2).
- **BW Channels**: The amount of single component channels (1 or 2).



FLC

A FLC object has only one format specific option:

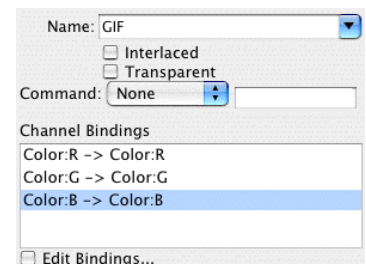
FLC Speed: The preferred playback speed as frames per second.



GIF

A GIF object has the following options:

- **Interlaced:** If set, the image data will not be stored linearly but in an interlaced manner.
- **Transparent:** If set, the black color will be set as the transparent color of the gif file. In later use of such a GIF image, pixels having the black color will appear transparent - the background shows through them.



ICO

The options of the Microsoft .ico format are:

Icon Type:

- 24 bit RGB - stores the colors
- 32 bit RGBA - stores the colors and the alpha channel
- Both 24 and 32 - this dual mode format includes both icon versions

FFMPEG

The FFMPEG options are:

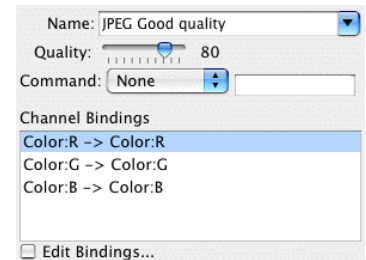
- **FPS:** Playback rate (frames per second)
- **Bitrate:** Data compression rate (kb/s). The higher the value, the better playback quality and bigger file size.
- **Format:** List of available formats, such as avi, mov or mpeg
- **Codec:** List of available codecs

Any codec can be used with any format but not all players support all format / codec combinations.

JPG

Jpeg files have only one format specific setting:

- **Quality:** Controls image compression and quality. The higher the quality, the bigger the output file size. High compression (=low quality) may cause bad errors in the image.



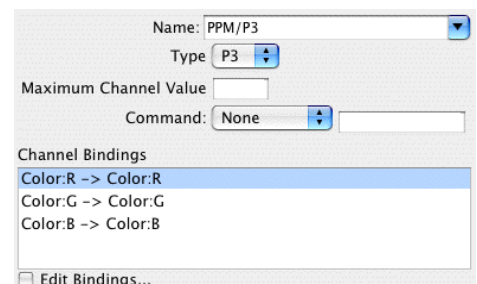
PNG

The PNG format has only one format specific option: a switch for turning Alpha channel on/off.

PPM

The PPM format has two special options:

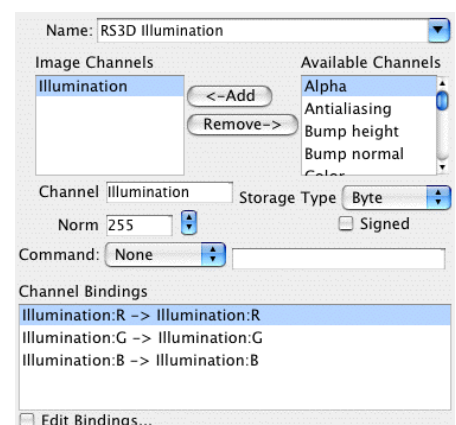
- **Type:** the PPM type selector (P3=ascii, P6=binary).
- **Maximum Channel Value:** Defines the integer value which represents the norm signal level 1.0 (white).



Realsoft 3D

This is the native file format of Realsoft 3D. It supports arbitrary amount of channels, and each sub channel can use any of the standard data types from a single BYTE up to 64 bit IEEE floating point format. This format implements practically unlimited data bandwidth. It is especially suitable for storing intermediate deep images which will be post processed later in high quality. The controls are:

- **Image Channels:** A list of channels that have been included into the image file format.
- **Available Channels:** A list of all rendering data channels available in the system.
- **Channel:** Name of the selected channel. You can give descriptive names to outputted channels, such as 'Floor alpha', 'Unshaded color' etc.
- **Storage Type:** Defines the data type of the selected image channel.



- **Norm:** Defines a conversion factor from floating point to integer form. The smaller the value, the wider value range the channel can store but with smaller precision. See the notes below.
- **Signed:** Enables/disables negative values for the channel.
- **Add:** Adds the selected channel from the system channels into the image format.

- **Remove:** Removes the selected image channel.



Note

For integer based data types (8 bit BYTE, 16 bit WORD, 32 bit INTEGER), the user can also define how floating point values are converted to the fixed point number form. The **Norm** value defines the conversion factor: integer value = float value*norm value. For example, for an usual byte based color channel, a suitable norm value is 255. Because this is also the largest value which 8 bits can represent, such a byte channel can only store float values between 0 - 1. Value -0.5 will be truncated to zero and 1.5 to 1. This is perfectly OK for the image color, but unsuitable for e.g. XYZ components of a surface normal vector.

Another related control is the **Signed** option. Turning it on makes it possible to store negative values to the channel. Channels themselves define convenient defaults for datatype management. Therefore, changing the Norm and Signed controls is seldom necessary.

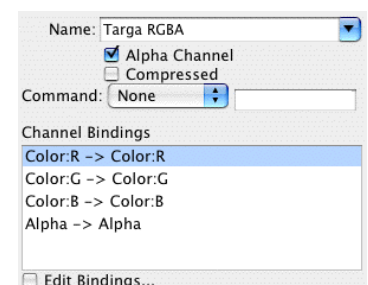
Some examples:

- The default norm value of a word based color channel is 4095. Because the largest value which a 16 bit unsigned word can store is 65535, the maximal signal which you can store to the channel is $65535/4095 = 16$. In other words, this kind of channel can store 16 times brighter values as the standard white signal (1,1,1). It is also $4095/255 = 16$ times more accurate as a byte channel. Summary: of the extra 8 bits of a word compared to a byte, 4 bits are used for the higher precision and 4 bits for the wider value range.
- If you wish to use a word channel to store the Z (distance) buffer and you know that the scene diameter is 100 meters, set the norm value to $65535/100 = 655$. Then you do not need to rescale distance values at rendering time to prevent data overflows.
- The suitable byte options for storing a surface normal are: Signed enabled, Norm value=127. X, Y and Z components of a unit vector run from -1.0 to 1.0, so the value range available must be split between negative and positive values. For example, you can store the bump normal needed for Global Illumination processing to a byte channel without noticeable loss of quality and save lots of memory.

Targa

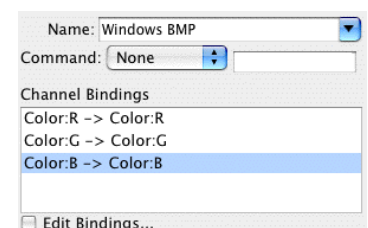
Targa is a popular output format in video industry because of its alpha channel support. It has two options:

- **Alpha Channel:** If set, an additional byte per pixel is allocated for storing the alpha values. Alpha channel usually contains the transparency information: value 0 = fully transparent, value 255 = fully non transparent.
- **Compressed:** Enables RLE compression, which can reduce the file size, especially when the image contains large constant color areas.



Windows BMP

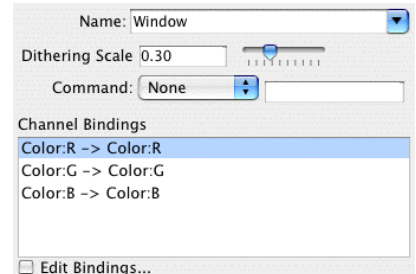
Windows BMP does not have any format specific options. It is the native image file format of the Windows operating systems and therefore supported by most applications, such as Windows Paint.



Window

The Window output is exceptional in that it is the only output target that is not disk file based. You can use it for previewing file rendering progress in real time. It has one option:

- **Dither Scale:** Defines the amount of RGB dithering. The dithering value is scaled automatically to match the bit depth of the current display mode. The window output specific dithering operates on the already byte converted data stream, and therefore the dithering accuracy is maximally 1/255 units. Therefore, it is irrelevant for 24 bit display modes. The default value 0.3 is optimal for 16-bit graphics.



Note

You can use render settings dithering for high accuracy dithering below 1/255 unit boundary.

Post Particle Effects

Lens Flare

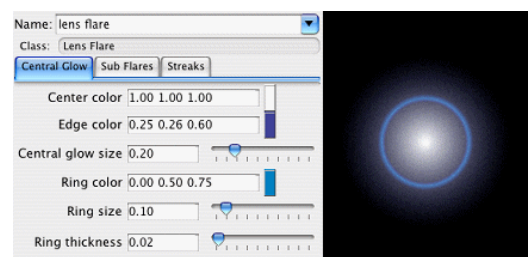
The 'Lens Flare' effect simulates optical phenomena caused by bright light in a camera's lens system. Lens flare has many controls, which allow for a wide variety of different effects. Several program modules utilize this effect:

- **Lens flare particle effect** (described below in detail). Particles define the position of flares.
- **Lens flare of a camera object.** Light sources, which illuminate the camera, create lens flares.
- **Image Flare effect.** Flares are added to the brightest areas of the image.

The lens flare controls are divided into three tabs.

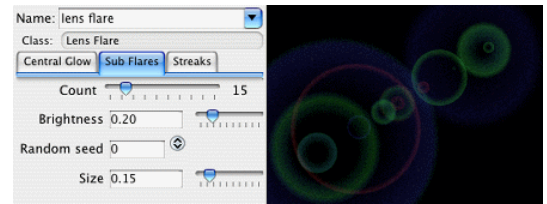
'Central Glow' tab:

- **Center color:** The color of the bright glow in the middle of the flare.
- **Edge color:** The central glow is faded away towards the edge through this color.
- **Central glow size:** The diameter of the central glow. Value 1.0 is the full image width.
- **Ring color:** The color of the thin ring around the central glow.
- **Ring size:** The diameter of the ring (1.0 is the same as image width). Set to zero to remove the ring.
- **Ring thickness:** The thickness of the ring.



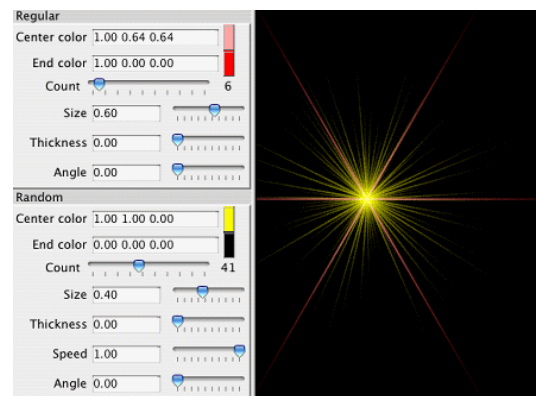
'Sub Flares' simulate internal lens reflections. The flares make a row passing through the center of the image.

- *Count*: The total number of circular sub flares. Set to zero to disable this part of the effect.
- *Brightness*: Defines how bright the sub flares are. Usually low values 0.05 - 0.2 give the best results.
- *Random seed*: Selects a sequence of random colors and sizes for the flares. Each seed value produces a completely different looking flare sequence.
- *Size*: The maximum diameter of sub flare rings.



The Streaks tab controls drawing of stars. Regular streaks have the same length and they are drawn at regular angle intervals. 'Random' streaks have varying lengths and random directions.

- *Regular & Random Center color*: The color of streaks in the middle of the flare.
- *Regular & Random End color*: The streaks fade away towards the end via this color.
- *Regular & Random Count*: The number of streaks.
- *Regular & Random Size*: The length of regular streaks and the maximal length of random streaks.
- *Regular & Random Thickness*: The thickness of streaks in the center of the flare.
- *Regular & Random Angle*: Defines the orientation of streaks. Changing the value rotates the streaks (which naturally can be animated).
- *Random Speed*: The random streaks appear and fade away in animations, creating an impression of light radiation. This control defines how quickly the streaks change. Value zero freezes the random streaks to be static.



Particle Disks

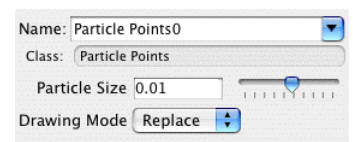
This effect draws a filled circle to represent each particle. The size of all the disks is constant and defined by the effect. The colors of the disks are defined by the particles' colors. You could, for example, texture map the particles using a 'Post Particles' shader (see the VSL manual for details). The edges of the rendered points are always faded to full transparency in order to guarantee smooth blending with the rendered image. Assigning a suitable 'Fade' channel value to the particles controls the overall transparency of the disks.

The effect is simple and quick to render when the disk size is small. Therefore it is quite suitable for representing smoke, dust, fire and other particle effects which can be defined by a large flock of particles.

The effect has the following controls:

- *Particle Size*: The common diameter of all particles. The size is measured as real world units; typically a value such as one millimeter is quite suitable. The effect scales the size of particles by taking the perspective projection properties of the camera into account. Zero size has a special treatment: each particle is drawn by one single pixel, which is a very fast operation.
- *Drawing Mode*: Defines how particles are drawn into the image. The drawing modes are listed below.

The drawing modes are:





'Add' sums the color of each new particle to the previous result. It is a suitable mode for drawing 'glowing' particles, because many overlapping particles produce a very bright result.



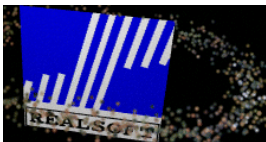
'Max' uses the maximum color of the previous result and the new particle. This method also produces a relatively bright result.



'Min' mode uses the minimum color of a particle disk and the background image. Therefore, the particles darken the image.



'Mul' mode multiplies the particle and background image colors. This usually darkens the image.



'Replace' mode is the default mode. It is suitable for representing non-glowing effects such as fog, smoke or dust.



Note

Particle disk positions can be randomized using the following kind of material:

```
Particle displacer material
    Post particles
        Coordinates += Noise(map coords)
```

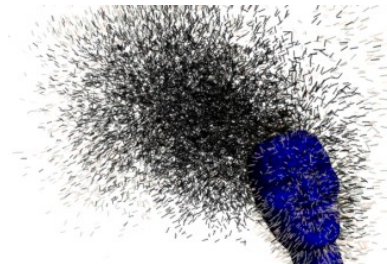
Particle Lines

This effect draws a line to represent each particle. The effect is therefore very suitable for rendering 2D particles (quickly falling raindrops etc.).

The thickness of the drawn lines is defined by the particle geometry. Color, fade and other rendering properties, as well as the drawing mode setting, are the same as in the Particle Points effect above.

The following channels can be modified using a Post particles shader to control particle line drawing:

- *Color* - particle color
- *Fade* - particle transparency
- *Coordinates* - start point of a particle
- *U direction* - end point of a particle
- *V direction X and Y* - two first sub channels define the radius values at the end points





Note

Mapping for the VSL material which assigns the above mentioned shader must appear before the line effect mapping in object hierarchy. Materials and post particle effects are evaluated in the order they appear in the select window.

Post Image Effects

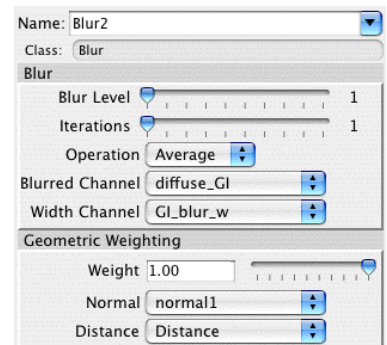
Blur Effect

The blur effect smoothes the image by blending adjacent pixels. This produces a smoother image and sometimes can be used to anti-alias the image. For example texture flickering of insufficiently anti-aliased animations can be reduced this way.

The Blur effect can be controlled using the following settings:

- *Blur Level*: Defines the amount of blur. Increasing the value makes images more blurred.
- *Iterations*: Repeats the effect multiple times. Unusually wide blurs can be created quickly using high iteration values.
- *Operation*: The performed pixel operation.
 - Average - Pixel's new value is the average of the values inside the blur filter
 - Min - Pixel's new value is the minimum of the values inside the blur filter
 - Max - Pixel's new value is the maximum of the values inside the blur filter
- *Blurred Channel*: The image channel to be blurred. Usually the color channel is the most suitable target.
- *Width Channel*: An optional channel for controlling blur size. For example, to obtain selective blurring, the user can add a channel property to part of scene objects and then apply blur to only those parts of the image where the channel values are set. The total blur size is computed as the following product:

$$\text{blur size} = \text{Iterations} * \text{Blur Level} * \text{Blur Width channel}$$
- *Geometric Weight*: If set to one, blurring happens only across connected, continuous surface areas. Dislocated objects will not blur together. This is a useful option when anti-aliasing object specific channel information (such as ambient lighting) by averaging in post processing phase.
- *Normal*: The channel, which defines surface normal continuity.
- *Distance*: The channel, which defines distance continuity.

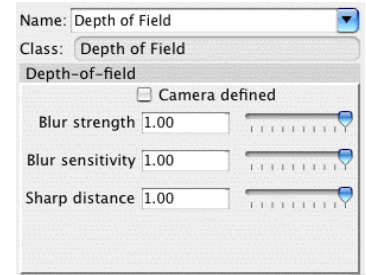


Depth of Field

The depth of field effect simulates image blurring of photographs. Cameras (except the pinhole camera) use a certain aperture for exposing the image to the film. Therefore, rays coming from different parts of the scene can expose the same point on the film. The greater the aperture, the more blur the photograph will contain.

The effect has the following controls:

- *Camera defined:* If set, the amount of depth of field blur is defined by the settings of the current camera (focal length, F-stop, focusing distance). Experienced photographers can estimate the amount of blur from the camera properties. Other users may find it more suitable to clear this option and use the other settings listed below to directly control the effect.



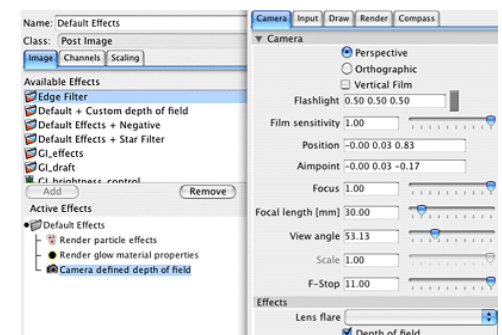
- *Blur strength:* Defines the maximum amount of blur when 'Camera defined' is off.
- *Blur sensitivity:* Defines how quickly the maximal amount of blur is achieved away from the sharp distance. Used only if 'Camera defined' is off.
- *Sharp distance:* The distance from the camera where the image is totally sharp (when 'Camera defined' is off).

Camera defined depth of field for view rendering

1. Go to the Post Image Effects tab. Drag and drop the post image you find suitable to the view window (for example 'Default Effects').

2. Open the property window to see the properties of the above mentioned post image. Make sure that a 'depth of field' object is included in the active effects. If not, add one to the active effects (you may first have to create a depth of field effect using the 'New' popup menu of the select window). Select the depth of field effect and verify from the property window that 'Camera defined' is set.

3. Open the view property window. Go to the Camera tab and set the Depth of Field option. Close the window and render the view. You can adjust the amount of blur from the F-Stop gadget of the view property window. The smaller the value, the more blur. Typical real world F-stop values range from 1.8 to 22.

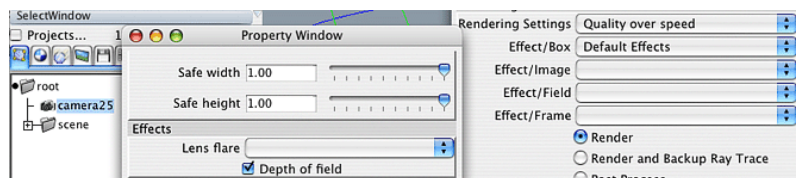


The post effect configuration includes Camera defined depth of field and the view has Depth of field enabled

Camera defined depth of field for file rendering

1. Go to the Post Image Effects tab. Select the post image you plan to use for file rendering (for example 'Default Effects'). Open the property window to see the properties of the post image. Make sure that a 'depth of field' object is included in the 'Active Effects'. If not, add one to the active effects. Select the depth of field effect and verify from the property window that 'Camera defined' is set.

2. Go to the Geometric Objects tab of the Select window. Make sure that there is a camera object included in the scene. If not, create one using the camera tool. Select the current camera object from the hierarchy (or all of them in case you are using multiple cameras for animation). Go to the Camera tab of the property window and set the 'Depth of Field' option. Close the property window.



The scene contains a camera, which has Depth of field option set. File rendering uses a post processing configuration Default Effects, which includes a Camera defined depth of field effect.

3. Open the file rendering window. Prepare settings as usual, but remember to select the post processing configuration which includes the depth of field effect. Render as usual.

Custom depth of field for view rendering

1. Go to the Post Image Effects tab. Drag and drop the post image you find suitable to the view window (for example 'Default Effects').
2. Open the property window to see the properties of the selected post image. Make sure that a 'depth of field' object is included in the 'Active Effects'. If not, add one to the active effects (create a new one using the 'New' popup menu of the select window if necessary). Select the depth of field effect and clear the 'Camera defined' option using the property window.
3. Render the view. Note that a strong depth of field effect may take a long time to render, so it may be a good idea to use 'Render/Define render box' menu of the view popup menu to render suitable test areas only. Adjust the 'Depth of Field' attributes from the property window and re-render (or post process only if backup buffer is enabled) until the result is suitable.

Note that you do not have to pay attention to the view property window's 'Depth of field' option in this case.

Custom depth of field for file rendering

1. Go to the Post Image Effects tab. Select the post image you plan to use for file rendering (for example 'Default Effects'). Open the property window to see the properties of the post image. Make sure that a 'depth of field' object is included in the 'Active Effects'. If not, add one to the configuration. Select the depth of field effect and clear the 'Camera defined' option using the property window.
2. Open the file rendering window. Select the post processing configuration, which includes the depth of field effect prepared above, to the 'Effect/Box' field.
3. Render as usual.

Dithering

This effect adds dithering to the image. Most commonly used image standards use a limited amount of colors to represent images. The human eye is very accurate in detecting differences between colors and therefore even 16 million colors is not always enough. Dithering improves image quality by blending available colors.

The effect has following controls:

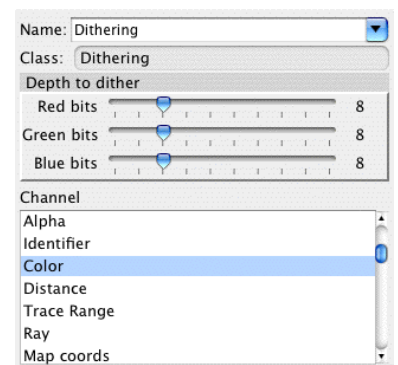
Red bits, Green bits, Blue bits: The suitable amount of dithering depends on the accuracy, which is used for storing the image. You can select the data type accuracy using these sliders. Value 8 is suitable for 24 bit output (16 million colors). Value 5 is suitable for 32 000 color output (16 bits).

Channel: The effect can be used for dithering any channel. Color channel is the usual target of dithering, but you can select another channel from this list.



Note

View window has its own dithering mechanism and the ray tracer output can also be dithered before post processing starts (while the data is still in accurate floating point form). Therefore dithering post effect is seldom needed.

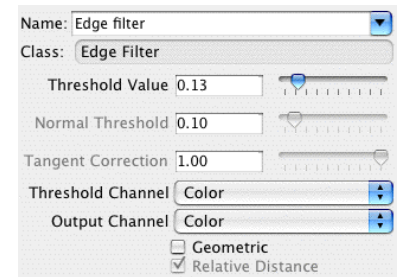


Edge Filter

The edge filter effect finds object silhouettes, edges and other areas where contents of the image change rapidly. The edge pixels are set black and other pixels white.

The controls of the effect are:

Threshold Value defines the amount of change between adjacent pixels which is required from edge pixels. The smaller the value, the more edges appear to the image rendered by the effect. A full value 1.0 usually generates an empty white image (when examining the color channel). In geometric filtering, the threshold value is applied to the Distance channel.

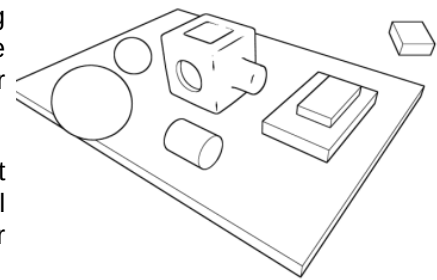


Normal Threshold is a second threshold which is used in geometric filtering. It defines the limit angle between two surface normals triggering edge drawing. Threshold value 1 corresponds to the 90 degree angle, 0 to zero angle. A suitably small threshold adds edges also to rounded but tightly bending corners.

Tangent Correction increases distance threshold of geometric filtering at low (tangential) viewing angles. The full correction value 1 is recommended for most scenes.

Threshold Channel selects the data channel, which is used for detecting edges. By default, color changes are examined. You can also use 'Distance' and 'Ray Normal' channels for detecting edges - or any other channel.

Output Channel defines to which channel the black and white result image is written. The default 'Color' channel overwrites the original shaded image. You can write the output to another channel and for example blend it with the shaded output using a VSL effect.



Geometric outlines

Geometric selects edge detection that is based on geometric continuity. Sharp angles and sudden distance changes generate edge pixels. Threshold channel will not be used.

Relative When enabled, distance discontinuity tests are made relative to the size of the scene. Furthermore, the threshold values are automatically adjusted by the output resolution and post image scaling. Distance and normal variations between two adjacent pixels are naturally smaller in a densely sampled high resolution image as in a low resolution preview image. This option automatically compensates the difference and helps to keep the result image similar across output resolution changes. Unfortunately identical detection cannot be guaranteed when resolution changes significantly.

Relative filtering is easier to set up, but the non-relative option provides more accurate control. For example, if a 5 mm thick plate on a table does not show up, reducing the Threshold to a value below 0.005 ensures that the edge will be detected.



Note

Surface normal based detection needs often some experimentation with the threshold value. An example: a small sphere viewed from a far distance may introduce significant normal direction change between every adjacent pixel. Edge filter which just sees the pixels cannot know if the surface is a 200 face polygonal sphere (=lots of edges) or a perfectly smooth round sphere (=no edges other than the round profile).

Fog

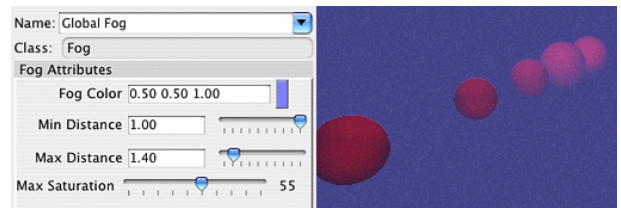
The Fog post processing effect provides a quick way to add global fog to the scene. The fog controls are:

Fog Color: This is the color of the fog.

Min Distance: Fog does not affect objects closer than this distance from the camera.

Max Distance: The distance from the camera which is needed for full fog effect. If saturation is 100, objects further than this distance are fully obscured by the fog.

Max Saturation: Defines how strongly the fog effect at its full level obscures objects. If the saturation is 100 %, the fog can fully hide objects. 50 % saturation blends maximally half of the fog color to the original image.

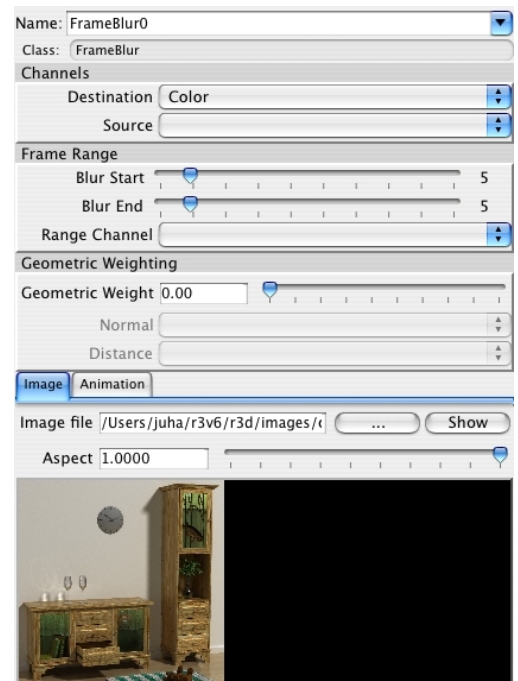


FrameBlur

The FrameBlur effect works similar to the normal Blur effect, except that it blurs over a sequence of images. This effect is useful for special effect purposes and for filtering image data over time.

The FrameBlur effect has the following controls:

- *Destination:* The post image channel to which the result is stored.
- *Source:* The image file channel to be blurred.
- *Blur Start:* Defines how many frames before the current image the blur starts. Value 1 starts the processing from the previous image.
- *Blur End:* Defines how many frames after the current image the blur ends. For example, if the current image index is 5 and Before=2, After=3, then the result will include the blurred result of images 3, 4, 5, 6, 7 and 8.
- *Range Channel:* An optional channel, which controls blur range. When a range channel is selected, the actual range is Before*range_channel ... After*range_channel. For example, if Before is set to 8 frames and the range control channel has a value 0.4 in a certain pixel of the current frame, blurring starts $8 \times 0.4 = 3$ frames before the current frame, provided that the respective pixels in preceeding frames have a range control value, which reaches the current frame. By setting the range channel to 0 or 1, one can obtain selective blurring i.e. only selected image areas will be blurred.
- *Geometric Weighting:* If set to one, blurring happens only across connected, continuous surface areas. Dislocated objects will not blur together. This is a useful option when anti-aliasing object specific channel information (such as ambient lighting) by averaging in post processing phase.
- *Normal:* The image file channel, which defines surface normal continuity for geometric blurring.
- *Distance:* The image file channel, which defines distance continuity for geometric blurring.



- **Image:** Here you can define an image file sequence (or avi file) from which the blurred data is read. The controls are the standard image object controls. If geometric weighting is active, the image file sequence must be in the native Realsoft 3D image format, and normal and distance channels must be included.



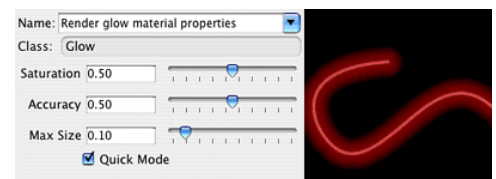
Note

The blur index range may exceed the actual source image range. The effect blurs the images it can find, and missing images do not cause an error. For example, at the end of the animation After indices usually refer to nonexisting images but that is perfectly OK.

Glow

The Glow effect is a part of the post effects 'Default Effects' configuration. It processes the image using two glow specific properties:

- Glow Size
- Glow Color

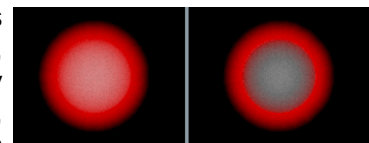


Glow effect

These can be defined just like any other material property such as color.

The glow effect examines the current scene and detects any use of the above mentioned channels. If they are not used, the effect is not executed. Therefore, you don't have to remove glow from the post processing configuration for speed reasons. You only have to remove it from the current post image in case the scene contains glow materials but you still want to disable glow rendering.

Saturation adjustment defines how quickly maximal glow brightness is reached in volume with respect to the distance. If the saturation value is zero, the glow looks brighter around the edges of the object - and has less intensity on the object itself, resulting in a kind of 'aura'. If the saturation is set to 1, an equally bright glow is generated in front of the glowing object as well as the edges.



Full saturation level on the left and zero saturation on the right

The **Accuracy** setting is an anti-aliasing type quality adjustment. The default value of 0.5 is suitable in most cases. The value range is from zero (fast, inaccurate) to 1.0 (slower but accurate).

The **Maximal Size** value defines a global limit for channel defined glow sizes. The global control can be used as an easy top level control for the effect, and to speed up the effect. Glow effect must usually render some additional image area outside the visible image to ensure effect continuity at image borders. The global size control instructs the program of what is the required extension area. The smaller the value, the quicker the rendering becomes. The default value is 0.1 (= 10 % of the image width).

Quick Mode uses a simpler solution to generate the glow from interior pixels of a glowing object. If the option is not set, every glow pixel generates a spherical 3D volume of a glowing matter around it, and such glows are blended into the final result. This is a very computation intensive task and takes long time to render. Quick mode uses a simpler approach, but usually the results are very similar - especially when glow size and color values variation is small.



Note

Suitable glow size values are typically quite small, for example 0.01 (1 centimeter) is pretty good for default scale models. Large glow sizes like 1.0 may take a long time to render.

When rendering an image with post effects to an animation, the rendering engine must expand the image internally. This increases rendering time significantly. If you only need to render one single still

image of a scene containing many glowing objects, you can often increase rendering speed dramatically by switching 'Post Proc./Safety Area' and 'Distrib/Box Rendering' off from the render settings property window. Alternatively, you may apply glow as a full image effect, not a box specific effect.

Gradient

The gradient filter computes the image gradient i.e. intensity change derivatives. The gradient is useful for image analysis and for special effects.

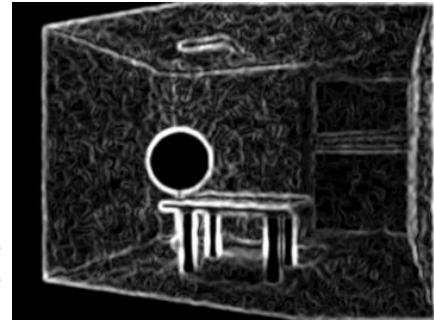
The Gradient effect has the following controls:

Brightness scales the brightness of the computed gradient data.

Input selects the data channel, from which the gradient is computed.

Output defines the channel, to which the gradient data is written.

Monochrome option outputs gradient data as a grayscale image representing the length of gradient values: $\sqrt{dx^2 + dy^2}$. If the option is cleared, dx is written to R component and dy is written to G component. Note that the value range of dx and dy usually includes negative values.



The gradient of noisy global illumination

Image Flare

The image flare effect finds the brightest areas of the image and adds highlight flares to them. It uses the following principles:

- Only places where a given threshold value is exceeded get a flare
- Only one flare is added per a continuously connected bright area
- The size of a flare depends on the brightness of the area generating it

Brightness Trigger defines the threshold value, which must be exceeded before a flare appears. The value is defined as a sum of red, green and blue. A pure white color has brightness 3.0 and pure red has brightness 1.0, etc.

Max Flare Count defines the maximum number of flares added to the image. Only the defined amount of brightest areas will get the flare. This is the maximum amount, not the exact amount. If the picture is too dark, no flares will be added.

Trigger Channel selects the channel which triggers flares. Usually it can be the color channel.

The lower part of the image flare properties window contains the standard lens flare gadget. See the description of the Lens Flare effect for details.

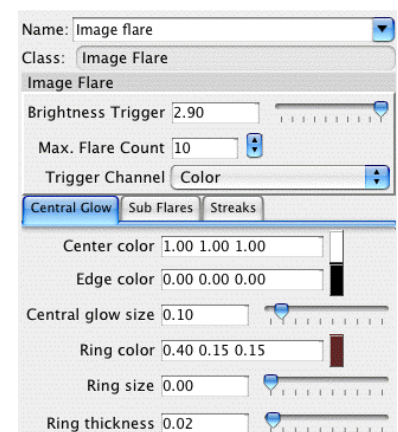


Image flare properties

The effect can extract highlight areas better, if you use 'Illumination' as the trigger channel. 'Illumination' value range is not clipped to 0..1 and its default data type is 'Word'. A byte can only represent a brightness scale from black = 0 to white = 1, whereas 'Word' channels can store values from zero up to 16. In other words, there are plenty of brightness values after white, and the program can locate the maximum brightness spots better.



Note

Do not use box rendering with the image flare effect, or apply image flare as an effect for full images. Render boxes break continuous bright areas and flare computations become inconsistent between boxes.

Light Glow

The Light Glow effect analyzes the image and generates an additive glow from bright image areas. Unlike the usual glow, light glow is a lens effect, meaning that it appears in front of all objects regardless of their distance from the camera.

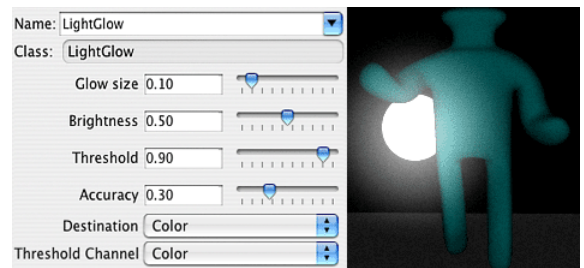
Light glow has the following controls:

Glow size -

Defines how wide the computed effect is.

Brightness -

Controls brightness of the overall effect. The intensity of the effect depends on the brightness of image areas generating the glow, and this slider provides a top level control to the whole effect.



Threshold -

Only image areas whose brightness exceeds this limit generate glow. Each sub channel is processed independently. If the 'Threshold' is 0.9 and pixel's illumination is red=0.5, green=0.5, blue=1.0, only blue glow becomes added to the surrounding area.

Accuracy -

Sampling accuracy for the effect. Values at 0.1 ... 0.5 range should work well for most cases. The full accuracy 1.0 can make the effect very slow, especially when the image contains thousands of pixels.

Destination -

The channel which glow changes. Usually should be 'Color'.

Threshold Channel -

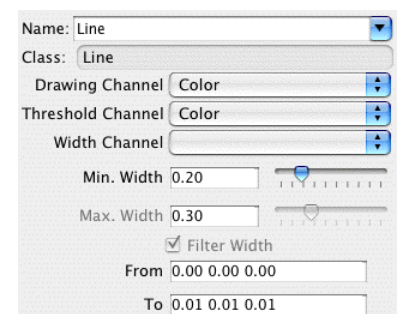
The channel whose brightness triggers the effect. This can also be 'Color'. However, since color channel values are usually limited to 0..1 range (because of byte storage type), the 'illumination' channel which has a higher dynamic range, allows more interesting effects (with the cost of increased memory usage).

Line Effect

The main purpose of the effect is to filter outline rendering. For example:

1. Ray trace in double resolution and outline rendering mode on
2. Apply Line effect to get thicker lines
3. Scale the result down to 50 % resolution

The result of this process is antialiased outline rendering



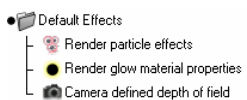
The following options are available:

- *Drawing Channel*: The channel to which lines are drawn
- *Threshold Channel*: The channel that triggers drawing
- *Width Channel*: An optional channel which can be used to achieve variable line width. The channel should have a value range between 0 and 1. Channel value 0 produces line width defined by the 'Min. Width' value and channel value 1 corresponds 'Max. Width' value respectively.
- *Minimal Line Width*: Line width values are expressed using percentage of the full image width. Therefore, small values such as 0.1 are usually suitable.
- *Maximal Line Width*: Used if width channel selected.
- *From and To*: Color range which triggers line drawing. For example, if you are rendering black outlines on a white background, the default range (0,0,0) - (0.01, 0.01, 0.01) is suitable. Then only almost black pixels will trigger the effect.

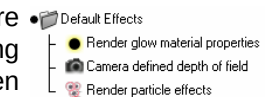
Particle Effect

This post processing object is actually not a real 'effect' - it does not render anything itself. However, it does the important job of managing the rendering of all post particle effects. The position of this object in the post processing configuration defines when post particle drawing takes place.

The configuration on the left shows a situation, where post particle effects are drawn as the first step of post processing. The right side image shows a configuration, which draws post particles last. Note that in the latter configuration, post particles cannot have the glow property.



The particle effect has an option for distance sorted drawing. When enabled, particles are drawn in distance sorted order. Sorted drawing is slightly slower than unsorted drawing (typically a couple of seconds per 100 000 objects). Sorting is usually necessary when rendering multi colored particle flocks.



Post Image

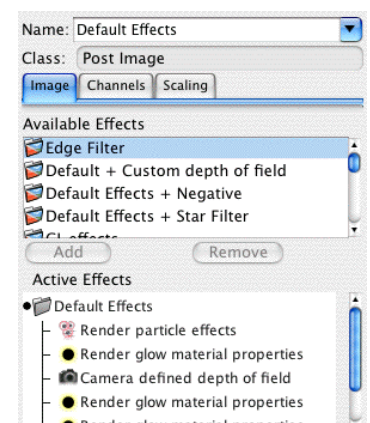
The Post Image object has an important role in all post processing configurations: it allocates the image buffer, which other post effects modify. It also acts as a parent effect, having a number of child effects, which modify the parent's image buffer.

A post image can be configured by adding a set of suitable effects to its Active Effects list. The child effects can be added by drag & drop from the Available Objects list or by using the Add popup menu item. Effects can be removed from the Active Effects list by selecting them and using the Remove popup menu item. The child effects are rendered in the order they appear in the Active Effects list - sometimes this has importance.



Note

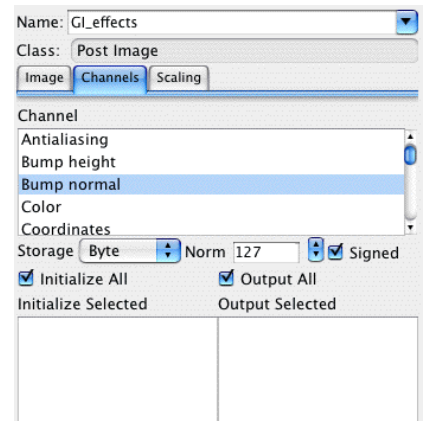
A component of a post effect configuration can be disabled temporarily by selecting it and using the Disable popup menu.



A post processing configuration defined by a post image

The Channels tab of the post image property gadget contains a list of all installed channels. When you select a channel from the upper Channel list, the Storage field displays the data type, which is used for storing the channel values. Byte takes the smallest amount of memory space but is inaccurate. Float is the most memory hungry but very accurate. The most suitable data type depends on the channel - Byte is usually enough for storing the color, but never accurate enough for the distance channel and cannot represent a wide enough value range. The default types are suitable for most cases, but sometimes using higher accuracy is necessary (see the image flare documentation for an example of this).

Float data type usually gives fastest rendering times, because it is the render engine's native way of representing channel data. Use of other data types leads to data type conversions. However, due to high memory consumption, it is seldom possible to use the float data type for all channels.



Channel controls

For integer based data types (8 bit BYTE, 16 bit WORD, 32 bit INTEGER), the user can also define how floating point values are converted to the fixed point number form. The **Norm** value defines the conversion factor: integer value = float value*norm value. For example, for an usual byte based color channel, a suitable norm value is 255. Because this is also the largest value which 8 bits can represent, such a byte channel can only store float values between 0 - 1. Value -0.5 will be truncated to zero and 1.5 to 1. This is perfectly OK for the image color, but unsuitable for e.g. XYZ components of a surface normal vector.

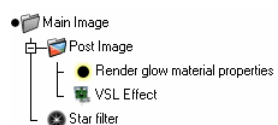
Another related control is the **Signed** option. Turning it on makes it possible to store negative values to the channel.

All channels define convenient defaults for these controls, so changing them is rarely necessary. Below are some examples:

- The default norm value of a word based color channel is 4095. Because the largest value which a 16 bit unsigned word can store is 65535, the maximal signal which you can store to the channel is $65535/4095 = 16$. In other words, this kind of channel can store 16 times brighter values as the standard white signal (1,1,1). It is also $4095/255 = 16$ times more accurate as a byte channel. Summary: of the extra 8 bits of a word compared to a byte, 4 bits are used for the higher precision and 4 bits for the wider value range.
- If you wish to use a word channel to store the Z (distance) buffer and you know that the scene diameter is 100 meters, set the norm value to $65535/100 = 655$. Then you do not need to rescale distance values at rendering time to prevent data overflows.
- The suitable byte options for storing a surface normal are: Signed enabled, Norm value=127. X, Y and Z components of a unit vector run from -1.0 to 1.0, so the value range available must be split between negative and positive values. For example, you can store the bump normal needed for Global Illumination processing to a byte channel without noticeable loss of quality and save lots of memory.

You do not have to define which channels to include in a post image. The post image can find out which channels are used by child effects. It includes them automatically.

Post images can be used hierarchically. Each effect must be added to a post image, then the post image can be added to another post image to obtain the hierarchy. The example image shows a simple hierarchical post image configuration.



Both 'Main image' and 'Post Image' are post image objects. The sub image 'Post Image' allocates a new image buffer and glow effect is rendered into it. The VSL effect does some custom manipulations after that. The final step is star filter effect rendering into the main image.

If *Initialize All* option (see the channel tab image above) is set, the post image automatically reads the contents of all the channels being used by its child effects from the parent image. In other words, when the effect drawing of the

sub post image starts, the situation is exactly the same as in the parent image. You can clear the Initialize All option and define manually which channels should be initialized in the beginning. Other channels will have zero contents. You can add channels to the Initialize Selected list by using the popup menu of the Channel list gadget. Removing channels happens by using the popup menu of the Initialize Selected gadget.

If the *Output All* option is set, all included channels are written to the parent image after sub effect rendering. To overrule this, clear the option and build a list of output channels to the Output Selected list.

Sometimes it is useful to clear the Initialize All option, but leave the initialization list empty. The effects will be drawn over a black background and the result can be, for example, added to the parent image. Anyway, it never makes sense to disable Output All and leave the output list empty - rendering effects without outputting the result somewhere just wastes rendering time.

The *Scaling* tab contains four values for scaling image data. Input scale X and Y rescale the incoming data, and Output Scale X and Y rescale the post processed data before outputting it further.

Examples:

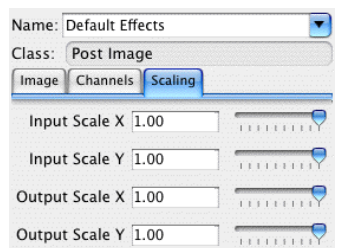
1. If both input scales are 2.0 and output scales are 0.5, the post image doubles the resolution before the post effects are executed. Since these input and output scales compensate each other ($2.0 * 0.5 = 1$), the scalings have no effect outside the post image in question.

2. If input scales of the top level post image are 1.0 and output scales are 0.5, also the ray tracing resolution is doubled. The resolution drops to half after post processing. This can be used as an antialiasing solution which covers all post effects. Ray tracing antialiasing level can be reduced correspondingly.

Note, that post images can be stacked hierarchically, and therefore you can change resolution between execution of each post effect.

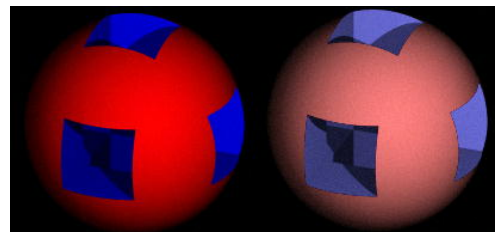
The *Recommended Use* gadget tells how a post effect configuration should be applied: per box, per image, per field or per frame. The fifth alternative is Not Defined, which usually means that the effect can be applied per box or per image equally efficiently.

The usage recommendation provides information for the user, who is selecting a post effect configuration manually. Secondly, the automatic post effect selection feature at project loading uses the recommendation, when selecting the post effect configuration for each view window from a loaded project.



Saturation

This effect changes the color saturation of images. It has only one property 'Color Saturation'. The lower the saturation level, the less colorful the processed image becomes. Full saturation 1.0 leaves the image unchanged, whereas zero saturation turns it into a gray scale image. Saturation has a maximum value of 1.0.



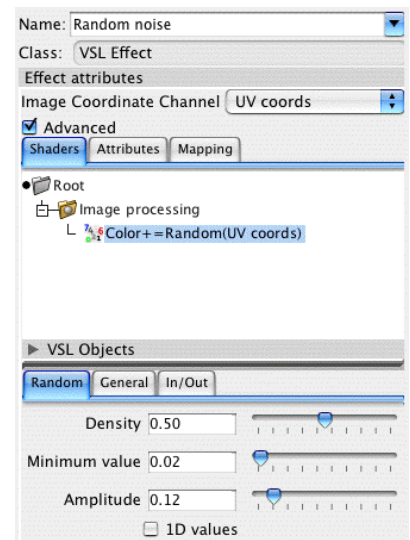
Saturation level 1.0 on the left, saturation 0.5 on the right

VSL Effect

The VSL Effect can be used for defining arbitrary pixel effects. Using the VSL shading language, the user can move data from a channel to another one, blend channels together, make color corrections, etc.

The **Image Coordinate Channel** defines the channel, which the effect uses to identify the coordinates of the processed pixel. The pixel coordinates are in floating point form: the top left corner has the coordinates (0,0,0) and the bottom right corner is (1,1,0). Use of relative floating point coordinates guarantees that VSL effects are resolution independent. When rendering boxes, the pixel coordinates are scaled to the range, which the box covers of the total image. Therefore, you can safely use box rendering with this effect.

The default image coordinate channel is the 'UV coords' channel. In the example image above, a random noise is added to the rendered image. The amount of noise depends on the UV coords.



The lower part of the VSL effect's property window contains the standard VSL editor. By checking the Advanced option, you get access to the editing tools. See the Materials chapter for VSL programming instructions.

The VSL objects of the image processing tree see all channels of the system and can freely perform any suitable computations using the channels. Because of the auto-configuration property of the post processing system, referring a channel from a VSL object automatically includes the channel in the post processing data flow.

Rendering Settings

A *Rendering Settings* object is a collection of rendering options. Realsoft 3D's photorealistic rendering engine is extremely powerful and designed for maximum realism and image quality. Correspondingly, rendering settings include a large number of options for controlling the entire rendering process.

Rendering settings can be managed through the select window's Render Settings tab. This tab shows the current library of predefined settings, and you can freely add new ones to suit your needs.

To create a new rendering settings object, select the 'New' pop-up menu. This creates a new object into the select window. As usual, you can use the property window to modify properties of the newly created object.

The rendering engine of Realsoft 3D consists of several sub-systems, such as the ray tracer, the post processor and scan line rendering. The first three tabs of the property window reflect this subdivision. Other tabs include general rendering options. All the available controls are shortly described below.

Ray Tracing

You can control ray tracing settings through the 'Ray Tracing' tab. Ray tracing is the most important part of the rendering pipeline. Accordingly, this tab contains many controls.

The controls are:

Detect

All the options in this frame are set ON by default. The rendering engine examines the scene and switches off internally all unnecessary computations. Therefore, none of these options slow down the rendering, unless the corresponding feature is really used in the scene. These options can be temporarily switched off to speed up preview rendering, for example.

Lighting

If set, the rendering engine computes lighting information. By resetting this check box, one can disable all light sources in the scene.

Shadows

If set, the rendering engine computes shadows.

Volume Shadows

If set, object interiors with suitable volumetric properties cast shadows. For example, a fog cloud can cast a shadow. Not the surface but the actual 'material' inside the volume casts shadows.

This is an advanced feature. The *Volume Filter* shader of VSL can be used for defining shadow casting volumetric materials.

Volumetric Effects

This option enables computation of fog, smoke, gas, plasma, etc. effects (except volume shadows which has a dedicated control as described above).

Lighting in Volume

If set, influence of light sources to fog like materials is computed. Some foggy materials may not be affected by light (self illuminating fog), whereas others glow in light. The latter type requires the use of this option to render.

Recursion

The basic principle of ray tracing is to follow a light source as it travels in a scene, hits a surface, becomes reflected to a new direction, hits another surface, etc. This process requires time consuming computations and must be stopped after a reasonable amount of steps. Otherwise it may happen that the program ends up following a light ray bouncing between two mirrors forever and the image never gets ready.

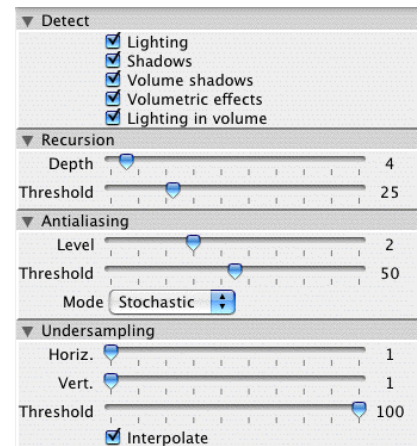
The settings for controlling the amount of steps are:

Depth

This value defines the amount of surface hits after which tracing of a ray stops. If the value is one, no reflections are rendered. Because the tracing stops as soon as the ray from camera (observer's eye) hits an object. A reasonable value range for photorealistic rendering is from 3 to 6. The minimal value that is required for a scene of an ordinary glass bottle is 5.

Threshold

Defines the minimal signal strength, which triggers a new ray. You can instruct the ray tracer to ignore very weak reflections or refractions not having much effect on the resulting image. This speeds up rendering. The threshold value is measured in per mill points: the value 100 is 10% of pure white color. Low values improve image quality but also increase rendering time.



Ray tracing settings



Note

When rendering transparent objects, this control can have a tremendous impact on the rendering speed. The ray is divided into two sub rays in each surface hit: the penetrating and reflected parts. Therefore, when recursion depth is increased, the amount of required computations grows exponentially, unless weak rays are terminated by a suitable threshold value.

Anti-aliasing

Ray tracer's anti-aliasing is a program feature for improving image quality. Without anti-aliasing, the ray tracer analyzes the contents of each pixel by computing one sample. Anti-aliasing can raise the amount of samples per pixel to ten or even more. Usually the additional samples are taken 'intelligently': the sampling rate is increased if the image data seems to vary a lot. This principle is called *adaptive anti-aliasing*.

Level

Controls the maximum amount of samples per pixel. The higher the value, the better the quality. Level 3 antialiasing is usually sufficient together with geometric anti-aliasing (see the modes below). Stochastic anti-aliasing may require one or two steps higher anti-aliasing level; level 4 is a good default for good quality rendering purposes.

Threshold

This value controls the adaptive nature of anti-aliasing. The ray tracer examines samples around the current sample. If the difference is higher than the threshold value, more samples are computed. Therefore, the smaller the threshold, the better the rendering quality. The value is expressed in per mill points: threshold 50 means that anti-aliasing starts if there is a color difference greater than (0.05, 0.05, 0.05) or (13, 13, 13) in 8-bit RGB scale. A good default range is between 50 and 100.

Zero level threshold can be used in conjunction with high stochastic anti-aliasing (level 6) to force maximal antialiasing quality.

Mode

Geometric: additional samples are taken along a regular sub grid of the pixel grid. Samples are used to find the areas where changes happen. This is a good and fast way to anti-alias images with reasonably low information density (clear geometric forms, etc.)

Stochastic: additional samples are taken randomly inside the pixel. Slower than geometric, but more reliable when the scene contains dense textures, thin objects, etc.



Note

Use stochastic mode for rendering motion blurred animations. Motion blur and stochastic anti-aliasing are designed to work well together.

Undersampling

Undersampling is quite a similar feature to anti-aliasing. It works as follows: A block of adjacent pixels is examined. If the color differences exceed a given threshold, the interior pixels are computed more accurately. Otherwise the color of the interior pixels is interpolated from the edge pixels.

This option is especially useful when rendering previews or high resolution images for printing purposes. Undersampling of 2*2 pixels is usually accurate enough to detect all details of a 2000*2000 pixel image and makes rendering 2-4 times faster.

Horizontal

The width of the pixel group. The higher the value, the faster rendering. This may reduce rendering quality, though.

Vertical

The height of the 'undersampled' pixel group.

Sampling Threshold

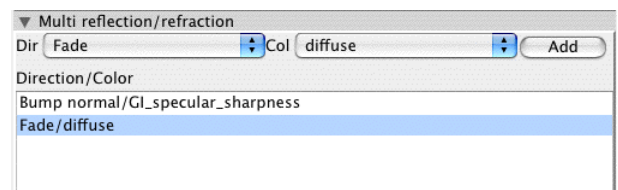
The minimal color difference (per mill points) which triggers more accurate sampling. The smaller the value, the better image quality. A threshold value 50 is sufficient for reasonable quality.

Interpolate

Activates interpolation of pixel groups where the threshold is not exceeded. If the option is inactive, undersampling generates mosaic like effects (for example, you may try sampling threshold 100 with no interpolation).

Multi Reflection/Refraction

Using controls on this frame, you can create unusual or customized ray tracing effects. For example: some real world materials create a double refraction. A lens of such material shows two different images. You can simulate this (or triple/quad, etc. refraction if you like) in Realsoft 3D.



Defining multiple reflections

Use the controls in the following way:

- Select a channel defining the direction of custom rays to the 'Dir' field. This is usually a user defined custom channel.
- Select a channel defining the amount of custom reflection/refraction to the 'Col' field. This may also be a custom channel.
- Press 'Add' button.
- You can select and remove undesired direction/color pairs from the list using the popup menu.
- Create a VSL material, which computes the direction and the color to the selected channels in the 'Surface properties' shader. The direction must be a unit vector. Black color eliminates the effect and white enables the effect at its full intensity.

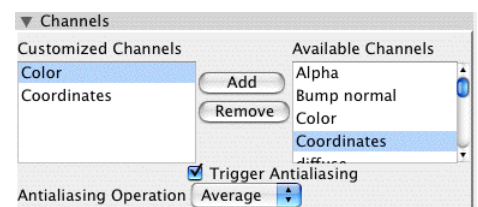
The example project 'tutorprojects/rendering/blurredreflections' shows an example of this feature. Four additional rays are cast randomly around the ideal reflection direction. This simulates a rough surface, which blurs reflections. Thanks to five reflection rays, anti-aliasing of blurred reflections is quite good.

Channels

The settings on this frame define data channel usage on anti-aliasing.

Trigger Antialiasing

By default, only color channel triggers antialiasing and subsampling. Here you can add also other channels that will trigger antialiasing. Especially alpha channel should be included whenever the alpha channel is outputted.



Anti-aliasing channels

Antialiasing Operation

Antialias operation selector. By default, all channels are averaged in antialiasing. Here you can change this to 'min' or 'max' operation. For example, setting the 'Distance' channel operation to 'Min' improves image quality when fog or depth of field post effects are used.

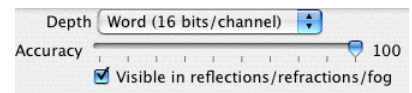
Scanline Rendering

Scanline rendering is the first step of the rendering process. Scanline particles are rendered to an internal 'meta image', which contains the properties of particles in a form which is suitable for the ray tracer. The ray tracer then combines scanline data to other scene data. Scanline rendering takes place before ray tracing and therefore scanline particles can be seen through glass objects, etc.

Scanline is a suitable rendering method for thin and small objects. Ray tracer can miss small details because of its 'sampling' principle.

Depth

Scanline particles define useful UV coordinates. This setting defines how accurately UV coordinates are stored per pixel. The default depth is 16 bit 'Word', but even 8-bit 'Byte' is sufficient for most cases. Low depth saves memory.



Scanline rendering options

Accuracy

The accuracy of the 'meta image'. Values lower than 100 % can be used for previewing purposes or to save memory.

Visible in Reflections/Refractions/Fog

If this option is set, scanline rendered objects are considered in ray tracing computations. Clearing the option may speed up the rendering.

Post Processing

Post processing is the last step in the rendering pipeline. It has the following settings:

Dithering

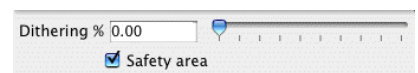
A general dithering, which is applied to the accurate 64-bit/channel floating point output from the ray tracer. The human eye is so accurate that it can see banding artifacts in slowly changing color gradients when the usual 24 bit RGB image representation (= 16 million colors) is used. Dithering improves quality by removing the banding. A suitable amount of dithering for a 24-bit image is $1/256 = 0.4\%$.

Note that some output targets (such as a view window) have their own internal dithering systems. In such a case, the additional dithering is unnecessary. This dithering feature is mainly used with file rendering.

Safety Area

If set, the post processing system uses a larger internal buffer to detect how 'out of camera view' effects affect the visible image. This option has significance only when post effects, which spread their effect over the image area, are being used (glow, depth of field, post particles). For example, the center of a lens flare can be outside the view, but some of the streaks can reach the view. If *Safety Area* is not checked, streaks would not be rendered.

If box rendering is not used and all objects fit inside the camera view (during the whole animation), the option can be switched off. The speed increase can be quite significant.



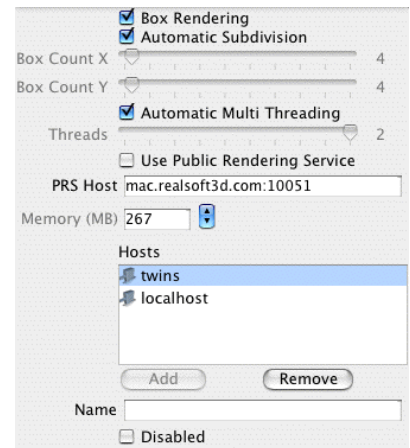
Distributed Rendering

The options of this tab control distributed rendering. Realsoft 3D can take advantage of processor power available in the workstation in question (Symmetric Multi Processing) as well as other workstations accessible through the network.

Box Rendering

This option activates distributed rendering. If checked, the rendering engine subdivides the image to a given number of sub images. If the rendered scene is complex, this may increase rendering speed. Box rendering adds some overhead and therefore rendering of very simple scenes may be slightly slower than without boxes. Box rendering saves memory (especially when scanline particles and post effects are used).

Box rendering should be set whenever multiple processors or network rendering is used to accelerate rendering.



Automatic Multi threading

Enable / disable automatic thread count detection. If checked, the distributed rendering system will use all the available processors for rendering. The higher the thread count, the higher the memory consumption.

Threads

The number of threads to be used for rendering. For example, if you have a dual processor system, set this slider to value 2. Single processor systems should use one thread. Zero threads can be used if the 'Hosts' list is not empty. The whole rendering is then performed using other computers on the network.

Automatic Subdivision

Enable / disable automatic box count detection. If checked, the distributed rendering system computes appropriate box count automatically based on the number of threads and sub servers used.

Box Count X/Y

These two sliders define the number of boxes in horizontal and vertical directions. The total number of boxes should be higher than the number of threads and hosts used for rendering. If all the systems of the network are equally fast, the box count can match the sum of hosts and threads. (e.g. 6 processors - set Box X to 1 and Y to 6). Higher box counts help the system to adapt to speed differences between processors: slow computers render only one box while fast computers render many. However, box rendering naturally involves some internal managing work, and at some point increasing box count starts decreasing the speed.

Automatic Network Rendering

Enable / disable automatic network rendering. If checked, the distributed rendering system uses all available rendering servers in your local network.

Note that this option won't use the rendering server running on the same system as the actual client program. The reason for this is that the software does not need network rendering to take advantage of the local CPUs.

Memory

If *Automatic Network Rendering* option is used, this field can be used for specifying minimum memory requirement for the servers to be used. Only those servers which have more memory than specified by this field will be used.

Hosts

Host names or IP addresses of machines on the network to be used for distributed rendering.

Remove/Add

These buttons allow you to add/remove hosts to the host list.

Disabled

Excludes the selected host from network rendering.

Outline Rendering

Outline rendering detects the edges (silhouettes) of objects and modifies the output accordingly. This feature can be used for technical representations, cartoon like rendering, etc. The following outline options are available:

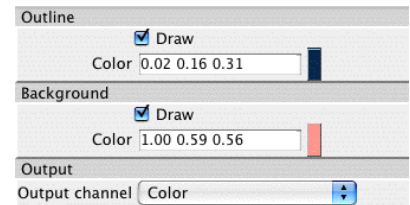
Outline/Draw - Activates outline rendering. Edges become drawn with a given color.

Outline/Color - Draw color of outline Edges.

Background/Draw - If set, interior of objects are filled with the given color. If cleared, the usual shaded surfaces fill the outline curves.

Background/Color - Background fill color.

Output channel - Defines the channel, to which the outline rendering is written. The default is color channel, which overwrites the usual shaded output. It is possible to direct the output to another (for example a user defined) channel and blend the outline and shaded output later using a suitable post effect.

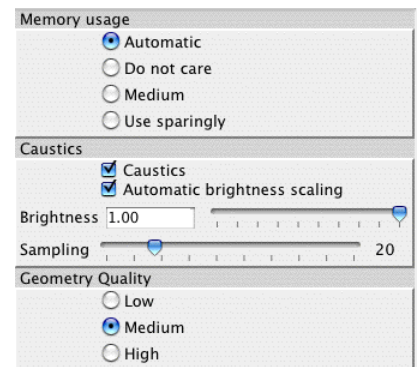


Misc

Memory usage frame:

The rendering engine of Realsoft 3D uses RAM memory to store intermediate computation results and optimization data in order to speed up rendering. When running multiple applications, this kind of memory usage may be undesirable. Also, if the scene is so complex that the operating system must use virtual memory, using low memory rendering methods can actually increase rendering speed.

You can select one of the four memory usage alternatives:



- *Automatic* - Render engine examines the system and tries to select the optimal memory usage.
- *Do not care* - Render engine uses RAM memory without limitations.
- *Medium* - RAM memory is used in a reasonable way.
- *Use sparingly* - The render engine uses as little memory as possible.



Note

Automatic memory usage is the easy and recommended choice. However, it is impossible to predict the exact memory demand in advance. Therefore, it is possible that rendering speed can be increased by forcing a particular memory mode instead of letting the render engine to estimate it. For example, if the automatic mode leaves lots of unused memory, 'Do not care' mode may be a faster solution. Or, if the automatic mode leads to intensive hard disk swapping, 'Use sparingly' mode may render faster. In a modern computer with several hundreds of megabytes of memory, this consideration is relevant

only for complex scenes (50 000 surface items or more). The automatic mode is always optimal for simple scenes.

Caustics frame:

Caustics rendering system analyzes how light rays originating from light sources travel and scatter across the scene. Caustics rendering is automatically activated by geometric objects which have caustics map feature enabled. The settings of this frame control caustics rendering.

Caustics

Turns Caustics rendering on/off. It may be necessary to switch Caustics off temporarily for draft rendering purposes.

Automatic brightness scaling

If set, the brightness of caustics effect is calibrated so that the maximal point of brightness has the value defined by the 'Brightness' slider below.

Brightness

Controls the brightness of caustics effects.

Sampling

Defines how accurately caustics is computed. The higher the value, the longer caustics rendering takes, but the more accurate the result is.

Geometry Quality frame:

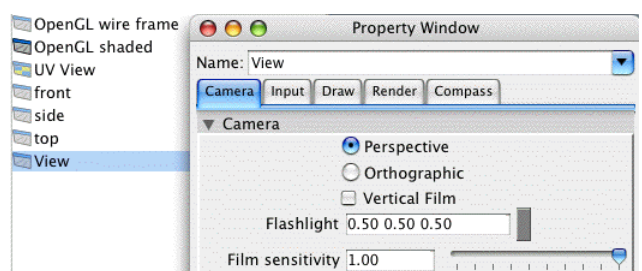
Many object types, such as SDS and NURBS, have adjustable render quality. The quality setting usually defines tessellation density for scanline drawing or accuracy of a numerical rayhit solving algorithm. The higher the quality, the smoother the surface appears.

The Geometry Quality option of render settings fine tunes object specific render qualities. For example, draft quality settings may use a lower accuracy to obtain faster rendering.

Medium quality renders with the actual quality value of each object. Low quality decreases object quality values and High quality increases the values. The actual amount of the relative change depends on the geometry type.

View Models

A View Model object contains all settings for a view window. You can assign a view model to a view by dragging and dropping it from the select window to the view. Each view window needs its own private view model. Therefore, if you assign a view model that is already in use to another view, the program automatically creates a duplicate and assigns it instead.



A view model object includes a large amount of options, which can be edited using the object property window and the view property window. For more information, see also Select Window/View Model Tab and View Window.

Animation Object

Animation object cannot be created directly. It is used by a number of other objects and provides a set of attributes for controlling timing. The attributes modify the local time and the effect propagates also to all the sub objects.

The object defines the following attributes:

Begin: Specifies the beginning of the object's time line.

End: Specifies the end of the object's time line.

Frequency: Scale factor for time. For example, setting Frequency to 2.0, will make the object run twice as fast.

Phase: Translation factor for time.

Periodic: Makes the time periodic, for creating loop animation effects.

Enable Time Curve: Enables the time deformation curve. If set, the time curve will be used for making objects local time non-linear.

Time Curve: The curve for deforming object's local time.



Note

If this feature is used, the key frame interface is no longer be able to show exact time values for the key frames!

Startup options

Realsoft 3D accepts the following startup options.

-tpaths [texture folders]

If a material refers to an image and the image is not found, these paths will be scanned to resolve the path. Useful if you have to render a project in another workstations. Multiple paths can be given by separating them with ; (semicolon).

Example: Realsoft 3D -tpaths "d:\mytextures;e:\moretextures"

-file [startup file]

Startup project.

-ccol [rrggbb]

Conflict color for the user interface controls representing conflicting attributes of multi selected objects.

-plugins [folder]

Specifies the folder from which the program reads plug-in modules. You don't usually have to specify this. The default values is 'Plugins'.

-lang [language]

This startup option allows you to select the language for the user interface. The value associated with the switch refers to sub folders of locale folder.

For example:

Realsoft 3D -lang english

Attempts to read strings from the folder Realsoft 3D\locale\english.

-port [port number]

Distributed network rendering system uses the port 10050. If you have already used this port for something else, or if you can't use this port for whatever other reason (firewall), you can change it with this option (in which case remember to tell the port also for RendD!).

Example:

Realsoft 3D -port 10099

-btree

If given, the select window uses balanced tree layout.

-noguiload

If specified, user interface specific data sections are not loaded from project files.

-render If given, the program renders the given project and then exits. Only the current frame is rendered. This option is useful if you have to render several projects from a script file.

Example:

Realsoft 3D -render -file proj1.r3d Realsoft 3D -render -file proj2.r3d Realsoft 3D -render -file proj3.r3d

-play

This option renders the project as an animation.

Example:

Realsoft 3D -play myanim.r3d

-firstframe [frame]-lastframe [frame]

These options can be used with **-play** option to define the frame range to be rendered.

Example: Let's assume you have two workstations and you want to render the frames 0 ... 100 in the first workstation and the frames 101 ... 199 in the second workstation.

In workstation 1 enter: Realsoft 3D -play -firstframe 0 -lastframe 100

In workstation 2 enter: Realsoft 3D -play -firstframe 101 -lastframe 199

-forcetoscreen

If given, the program moves all windows to a visible screen so that the user can see them. You may need this option when reading projects that has been created on a multi monitor system.

-home [folder]

Specifies the home directory for the local user files. You don't usually have to touch this.

-nethome [folder]

If Network setup is used, executable files, manuals and other user independent files are installed on a server. This option specifies the home directory for these common program files. You don't usually have to touch this.

-debug

If specified, the program prints an error message whenever it finds a request it doesn't recognize. Can be useful for tracking old plug-in modules down, for example.

-prshost [hostname]-prsport [port]

These startup options control distributed network rendering. Servers to be used for rendering can be specified explicitly, by entering them to the Hosts list of rendering settings window. Or, you can ask the program to find them automatically from the host specified by these startup options. The host referred must run the PRS service.

-helpdir [folder]

This parameter can be used for defining custom path for the reference manual.

-html

If passed, the program attempts to use the html help manual rather than the native help manual.

-stcanvas

Enable single threaded rendering model. Helps to run the program with "beta" graphics drivers. help.

Scripting

Built-in JavaScript Classes

Axis Class

Realsoft 3D Axis.

Axis object consists of two attributes: position and direction. It can be used, for example, for representing vertices and associated vertex normals.



pos - vector defining origin (position) dir - vector defining direction For example:

```
o = new r3Vect(0.2, 0, 0);  
d = new r3Vect(1, 0, 0);  
a = new r3Axis(o, d);
```

Methods

Method: r3Axis - constructor

Syntax: a = new r3Axis()

Parameters: vPos, vDir - two vectors specifying origin and direction

Returns: new axis object

Description: Creates a new axis object. If no parameters are passed to the constructor, a null axis system is created (both attributes will be null vectors). If the number of parameters is two, then the constructor assumes that the parameters are vectors defining position and direction.

Example: // create an identity axis

Method: identity - identity

Syntax: a.identity();

Parameters: -

Returns:	-
Description:	Reset axis to identity state. This corresponds to the following code: <code>a.pos.set(0, 0, 0); a.dir.set(0, 0, 0);</code>
Example:	<code>cs.identity();</code>
Method:	normalize - normalize dir vector
Syntax:	<code>a.normalize();</code>
Parameters:	-
Returns:	-
Description:	Normalize dir vector to unit vector. This corresponds to the following code: <code>a.dir.norm();</code>
Example:	<code>a.normalize();</code>
Method:	print - dump values
Syntax:	<code>a.print();</code>
Parameters:	-
Returns:	-
Description:	Print current values of the axis object. Useful for debugging purposes.
Example:	<code>a.print();</code>

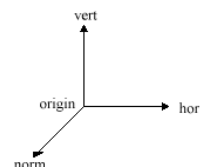
Coordinate System Class

Realsoft 3D Coordinate System.

Coordinate system is analogous to Matrices in that it can be used for representing transformations and spaces.

origin - vector defining origin hor, vert, norm - three axes For example:

```
o = new r3Vect(0.2, 0, 0);
x = new r3Vect(0.1, 0, 0);
y = new r3Vect(0, 0.1, 0);
z = new r3Vect(0, 0, 0.1);
cs = new r3Coordsys(o, x, y, z);
```



Methods

Method:	r3Coordsys - constructor
Syntax:	<code>v = new r3Coordsys()</code>
Parameters:	vOrigin, vHor, vVert, vNorm - four vectors specifying origin and axes for the coordinate system.
Returns:	cs - new coordinate system object
Description:	Creates a new coordinate system object. If you don't pass any parameters to the constructor, an identity coordinate system is created. If the number of parameters is four, then the constructor assumes that the parameters are vectors defining coordinate system's origin, horizontal, vertical and normal axes.

Example:	// create an identity cs ;
Method:	identity - identity
Syntax:	cs.identity();
Parameters:	-
Returns:	-
Description:	Reset coordinate system to identity state. This corresponds to the following code: cs.origin.set(0, 0, 0); cs.hor.set(1, 0, 0); cs.vert.set(0, 1, 0); cs.norm.set(0, 0, 1);
Example:	cs.identity();
Method:	normalize - normalize coordinate system axes
Syntax:	cs.normalize();
Parameters:	-
Returns:	-
Description:	Normalize coordinate system axes to unit vectors.
Example:	cs.normalize();
Method:	print - dump current coordinate system values
Syntax:	cs.print();
Parameters:	-
Returns:	-
Description:	Print current values of the coordinate system object. Useful for debugging purposes.
Example:	cs.print();

Low Level API

Scripting languages are linked to Realsoft 3D through a small set of low level functions corresponding to those used in the Realsoft 3D SDK 'C' API.



Note

The API described here is not intended for end users but for class implementators and other advanced users who need to get full control over Realsoft 3D.

Scripts can access Realsoft 3D through two variables:

- r3MainWindow - root of the user interface (main window)
- r3Model - root of the model (functionality).

These variables are actually 'pointers' which point to different type of objects depending on the application.

Functions

All the low level functions deal with Realsoft 3D objects rather than JavaScript objects. For sake of speed, there is no type checking. Passing wrong type of parameter to a function may crash the program. Type checking is only implemented by higher level JavaScript API.

Function:	include - include a JavaScript file
Syntax:	include(filename);
Parameters:	filename - JavaScript file to be included
Returns:	-
Description:	Loads in given JavaScript file, if not already loaded. This function corresponds to #include directive of 'C/C++' and can be used for loading JavaScript files defining "proto type" objects (in Java or C++, these would be called 'classes').
Example:	<pre>include("oops/r3button.js");</pre>
Function:	load - load a file
Syntax:	load(filename);
Parameters:	filename - the name of JavaScript file to be loaded
Returns:	-
Description:	Loads in given JavaScript file. Unlike include(), this function doesn't detect whether the file was already loaded. A file can be loaded any number of times.
Example:	<pre>load("myclasses/toolbars/mycreationtools.js");</pre>
Function:	MakeID - Create an IFF identifier
Syntax:	id = MakeID(a,b,c,d);
Parameters:	a,b,c,d - four ascii values
Returns:	32 bit IFF identifier
Description:	Creates a 32 bit IFF identifier from the given four 8 bit values.
Example:	<pre>var myid = MakeID('I','a','g','e');</pre>
Function:	R3New - create a new Realsoft 3D object
Syntax:	obj = new R3New(classid, taglist);
Parameters:	classid - class identifier
Returns:	obj - realsoft 3d object
Description:	Creates a new Realsoft 3D object and returns a handle to it. The type of the object to be created is defined by 'classid'. Properties (attributes) for the object can be defined by one or more tag - value pairs. Class id, methods and attributes for the object are defined in the object's class header file. All class header files can be found in 'scripts' folder.
Example:	<pre>// Create a sphere object include("real/objects/r3sphere.js");</pre>


```
sphere = R3New(R3CLID_SPHERE,
               [R3RA_Name, "my sphere",
                R3SPHA_Center, new r3Vect(0.0, 0.1, 0.0),
                R3SPHA_Radius, 0.1]);
```

Function: R3Dispose - delete object

Syntax: R3Dispose(obj);

Parameters: obj - realsoft 3d object to be disposed

Returns: -

Description: Deletes the given Realsoft 3D object

Example:

```
// create a sphere
sphere = R3New(R3CLID_SPHERE, ...);
...
// delete the sphere
R3Dispose(sphere);
```

Function: R3Set - set object attribute

Syntax: R3Set(obj, id, value, type, flags);

Parameters: obj - address of Realsoft 3D object

Returns: -

Description: Set attribute. This function corresponds to 'R3SetAttrs()' function of 'C' API. The parameter 'object' refers to Realsoft 3D object. The 'id' is a unique identifier specifying the attribute to be set. The type of the 'value' parameter **must** match the type of the attribute to be set. There is no run time type conversion and the type of the attribute must be correct or program may crash.

Example:

```
// move the main window to x=10, y=20
R3Set(r3MainWindow, R3WGA_Left, 10, R3TID_INTEGER, 0);
R3Set(r3MainWindow, R3WGA_Top, 20, R3TID_INTEGER, 0);
```

Function: R3Get - fetch object attribute

Syntax: R3Get(obj, id, value, typeid, flags);

Parameters: obj - Realsoft 3D object

Returns: value - fetched attribute.

Description: Fetch object attribute. This function corresponds to R3GetAttrs() function of 'C' API. Typeid parameter specifies the type of the attribute to be fetched. Supported types are defined in 'oops/r3typids.js'. The 'byvalue' attribute indicates whether the attribute is fetched 'by value' or 'by reference'. This is needed to tell R3Get() whether it should allocate temporary buffer or not.

Example:

```
name = R3Get(r3MainWindow, R3WA_Title, R3TID_STRING, R3TNS_COPYSTRING);
print("Current window title is " + name);
```

Function: R3Do - call a method

Syntax: rc = R3Do(obj, mth, taglist) - call a method

Parameters: obj - address of Realsoft 3D object

Returns: rc - method specific return value

Description: Send a method to Realsoft 3D object. There are six different functions which corresponds to those used by SDK/'C' API. Because JavaScript keeps track of number of passed parameters, there is no need to terminate the tag list by 'R3TAG_END'.

Example:

```
// Ask the Realsoft 3D to find out its native size
R3DoA(r3MainWindow, R3WGM_FIT, R3WFP_BESTFIT);
```

Function: R3ProcessEvents - start event processing

Syntax: R3ProcessEvents(obj);

Parameters: obj - main window

Returns: -

Description: Start event processing for the given window. There can only be one event loop per application. If there is already an event processing loop running on the application in question, the function returns immediately. Otherwise the function returns when the event loop is terminated.

Example: R3ProcessEvents(window);

Function: R3Exit - stop event processing

Syntax: R3Exit();

Parameters: -

Returns: -

Description: Terminates event processing loop, which most likely causes the program to exit.

Example:

```
R3Exit();
```

Function: R3ClassFind - find class

Syntax: obj = R3ClassFind(classid);

Parameters: classid - class identifier

Returns: obj - class object

Description: Find class by given id. If the class is not registered, returns NULL.

Example: In Realsoft Graphics Oy's object oriented programming system, constructor is a class method. Creating a sphere means we call R3RM_CREATE method of the sphere class.

```
include("real/objects/r3sphere.js");
```

```
// find sphere class
class = R3ClassFind(R3CLID_SPHERE);
if(class) {
    // ask sphere class to create sphere instance
    sphere = R3Do(class, R3RM_CREATE,
                  [R3SPHA_Radius, 0.1]);
}
```

Function: Class - show class info

Syntax: Class(name);

Parameters: name - name of the object

Returns: -

Description: Show class specific info.

Example:

```
Class("CurrentProject.Geometrics.Root");
```

Function: Attrs - list public attributes

Syntax: Attrs(name);

Parameters: name - hierarchical name specifying the object

Returns: -

Description: This function attempts to find the given object and enumerate its attributes. The name is a hierarchical name which identifies an object in the application in question.

Example:

```
// Ask a sphere to enumerate its attributes
Attrs("CurrentProject.Geometrics.Root.mylevel.mysphere");
```

Function: Get - fetch attribute by name

Syntax: value = Get(name);

Parameters: name - hierarchical name specifying a Realsoft 3D attribute to be fetched

Returns: obj - attribute

Description: Fetches an attribute by symbolic name.

Example:

```
// fetch the address of the project list object in Realsoft 3D
aProjectList = Get("");
```

```
// fetch the current project object
aCurrPrj = Get("CurrentProject");
```

```
// fetch the name of the current project
```

```
sName = Get("CurrentProject.Geometrics.Name");  
  
// fetch the name of the Root level  
sName = Get("CurrentProject.Geometrics.Root.Name");
```

Function: Set - set public attribute

Syntax: rc = Set(name, value);

Parameters: name = symbolic name identifying the object/attribute

Returns: -

Description: Set public attribute.

Example:

```
// rename the root level  
Set("CurrentProject.Geometrics.Root.Name", "my new root");  
  
// set the radius and center of 'mysphere'  
Set("CurrentProject.Geometrics.Root.mysphere.Radius", 0.2);  
Set("CurrentProject.Geometrics.Root.mysphere.Center", new r3Vect(0.2, 0.1, 0));  
  
// set current animation frame  
Set("CurrentProject.Animator.Frame", 30);
```

Function: Co - set Current Object

Syntax: ok = Co(name);

Parameters: name - symbolic name

Returns: ok - true if succeeded

Description: Make the given object the current working object. Names passed to Get(), Set() and Attrs() functions are relative to the *current object*.

Example:

```
// make the 'root' level the current working object  
Co("CurrentProject.Geometrics.Root");  
  
// change the root's name  
Set("Name", "my root level");
```

Function: Up - make the parent object the current object

Syntax: Up();

Parameters: -

Returns: -

Description: Make the parent object the current working object. Returns false if there is no parent object.

Example:

```
// make the 'root' level the current working object
Co("CurrentProject.Geometrics.Root");

// change the root's Translate attribute
Set("Translate", new r3Vect(0, 0, 0));

// pop up one level
Up();

// set root object's name
Set("Root.Name", "my root level");
```

Matrix Class

Realsoft 3D Matrix is a row dominated 4x4 matrix. Realsoft 3D uses matrices to represent spaces and transformations for geometric objects.

	a0	a1	a2	a3	
	b0	b1	b2	b3	
	c0	c1	c2	c3	
	d0	d1	d2	d3	

Matrix operations (such as concatenation, scale, translate etc. concatenate the new matrix on the right side. In other words, the latest operation is executed the first using 'first in last out' principle. a0, a1, a2, a3, b0, ... d3 - 16 floating point numbers defining 4 x 4 matrix. For example:

```
m.a0 = m.b1 = m.c2 = 1.0; // reset scale transformation component to identity
```

Methods

Method: r3Matrix - constructor

Syntax: v = new r3Matrix()

Parameters: a0, ... d3 - sixteen floating point numbers

Returns: m - new matrix object

Description: Creates a new matrix object. If you don't pass any parameters to the constructor, an identity matrix is created. If the number of parameters is four, then the constructor assumes that the parameters define a coordinate system's origin, horizontal, vertical and normal axes.

Example: // identity matrix;

Method: identity - identity

Syntax: v.identity();

Parameters:	-
Returns:	-
Description:	Make the matrix identity matrix
Example:	<code>v.identity();</code>
Method:	set - set matrix attributes
Syntax:	<code>m.set(a0, a1, ... d3);</code>
Parameters:	a0, ... - sixteen floating point numbers
Returns:	-
Description:	Set matrix
Example:	-
Method:	translate - concatenate translation transformation to the matrix
Syntax:	<code>m.translate(x, y, z);</code>
Parameters:	x, y, z - floating point numbers specifying translation in x, y and z dimensions
Returns:	-
Description:	Concatenates given translation vector to the matrix.
Example:	<code>m = new r3Matrix();</code>
Method:	scale - concatenate scale transformation to the matrix
Syntax:	<code>m.scale(fX, fY, fZ);</code>
Parameters:	vX, vY, vZ - scale factors in x, y and z dimensions.
Returns:	-
Description:	Concatenates scale transformation to the matrix. vOrigin parameter may be given to specify scale origin. vOrigin defines a point which is translated to origin before the actual scale is applied. If no scale origin is given, [0, 0, 0] is assumed.
Example:	<code>m = new r3Matrix();</code>
Method:	rotate - concatenate rotation transformation
Syntax:	<code>v.rotate(fAngle, vAxis);</code>
Parameters:	fAngle - rotation angle in radians
Returns:	-
Description:	Concatenates rotation specified by angle, origin and rotation axis to the matrix. The rotation angle is defined in radians. If no origin is specified, [0, 0, 0] is assumed.
Example:	<code>// rotate about the 'y' axis</code>
Method:	rotate2v - concatenate rotation defined by two vectors
Syntax:	<code>v.rotate2v(vFrom, vTo);</code>

Parameters: fOrigin - rotation origin

Returns: -

Description: Concatenates rotation specified by two vectors.

Example: // rotate 90 degrees about the 'y' axis

Method: skew - concatenate skew

Syntax: m.skew(fAngle, vFrom, vTo);

Parameters: fAngle - skewing angle in radians

Returns: -

Description: Concatenates skew transformation into the matrix. Skew transformation is defined in space defined by two vectors 'vFrom' and 'vTo'.



Note

The skewing angle must satisfy the following constraints:

$$-\text{PI}/2 < \text{angle} < \text{PI}/2$$

Example: // skewing space is x - y plane

Method: invert - inversion

Syntax: m.invert();

Parameters: -

Returns: -

Description: Invert the matrix. Assumes that the matrix is invertible.

Example: m.invert();

Method: mktran - make transformation matrix

Syntax: m3 = m.mktran(m2);

Parameters: m2 - matrix specifying target space

Returns: m3 - transformation matrix mapping points from 'm' to 'm2'

Description: Computes a matrix which maps a point from space defined by matrix 'm' to target space specified by 'm2'.

Example: mSrcSpace = new r3Matrix(o, h, v, n);

Method: transform - transform a vector

Syntax: m.transform(vTo, vFrom);

Parameters: vFrom - vector to be transformed

Returns: -

Description: Transform given vector. If two parameters are given, the method transforms the given 'vFrom' vector and assigns the result to 'vTo' (operation doesn't change 'vFrom'). If only one parameter is specified, the parameter acts as both source and target i.e. the result is assigned back to the given vector.

Example: // define a transformation matrix

Method: transforml - transform by 3 x 3 top-left sub-matrix

Syntax: m.transforml(vTo, vFrom);

Parameters: vFrom - vector to be transformed

Returns: -

Description: Transforms a vector by operating the parameter only with the 3*3 top-left submatrix, which is the linear part of the affine matrix operator. This means that you can use the transforml() method to transform vectors.

Example: // define a transformation matrix

Method: null - make the matrix null matrix

Syntax: m.null()

Parameters: -

Returns: -

Description: Sets all matrix components to null

Example: m.null();

Method: transpose - transpose matrix

Syntax: m.transpose()

Parameters: -

Returns: -

Description: Transpose matrix

Example: m.transpose()

Method: cmp - matrix comparison

Syntax: b = m.cmp(m2);

Parameters: m2 - matrix to be compared

Returns: b - boolean, true if matrices are identical.

Description: Compares the matrix 'm' against the given matrix 'm2' and returns true if the two matrices are identical.

Example: rc = m.cmp(m2);

Method: copy - copy a matrix

Syntax: m2 = m.copy();

Parameters:	-
Returns:	m2 - new matrix
Description:	Creates an identical copy of matrix.
Example:	m2 = m.copy();
Method:	orthogonal - make orthogonal projection matrix
Syntax:	m.orthogonal(fLeft, fRight, fTop, fBottom, fNear, fFar);
Parameters:	fLeft ... - minimum and maximum clipping planes
Returns:	-
Description:	Concatenates orghogonal projection to the matrix. Transformation maps fNear to 0 and fFar to 1.0
Example:	m.orthogonal(-0.1, 0.1, -0.1, 0.1, 0.1, 5.0)
Method:	perspective - concatenate perspective projection
Syntax:	m.perspective(fMinX, fMaxX, fMinY, fMaxY, fNear, fFar);
Parameters:	fMinX ... - clipping planes specifying perspective projection
Returns:	-
Description:	Concatenate perspective projection to the matrix. Projection maps 'fNear ... fFar' to '0 ... 1'
Example:	-

Realsoft 3D specific Classes

Realsoft 3D application consists of thousands of classes, which all can be accessed through JavaScript.

There is a JavaScript header file for each Realsoft 3D class. These so called JavaScript wrapper classes can be found in scripts/js folder. The folder structure corresponds to the structure of Realsoft 3D application.

There are three sub folders:

- oops - the kernel library (object oriented programming system)
- real - Realsoft 3D built-in libraries
- plugins - plug-in libraries

For more information about Realsoft 3D classes, see Realsoft 3D JavaScript Class Reference. You can find it from the Realsoft 3D CD. It is also online at [realsoft.com](http://www.realsoft.com) [<http://www.realsoft.com>]

Vector Class

Realsoft 3D defines two separate vector classes for managing three and four dimensional vectors. Vector classes are built-in classes for sake of speed. Three dimensional vector class defines three attributes: x, y, z: Four dimensional vector defines attributes x, y, z and w.

Methods

Method: r3Vect - constructor for 3d vector

Syntax: `v = new r3Vect()`

Parameters: x, y, z, w - floating point numbers

Returns: v - new vector object

Description: Creates a new vector object. If you don't pass any parameters to the constructor, a null vector is created i.e. all the vector attributes are initialized to zero.

Example:

```
v = new r3Vect();
v = new r3Vect(0.1, 0.2, 0.3);
```

Method: add - addition

Syntax: `v2 = v.add(v1);`

Parameters: v1 - vector

Returns: v2 = result vector

Description: This adds the given vector to the object in question and returns the result as a new vector. The operation doesn't change the attributes of the object.

Example:

```
v1 = new r3Vect(0.1, 0, 0);
v2 = new r3Vect(0.2, 0, 0);
v3 = v1.add(v2);
```

Method: fadd - vector addition

Syntax: `v.fadd(v2);`

Parameters: v2 - vector to be added

Returns: -

Description: Vector operation $v = v + v2$.

Example:

```
v = new r3Vect(1, 0, 0);
v2 = new r3Vect(0.1, 0, 0);
v.fadd(v2);
```

Method: set - set value

Syntax: `v.set(x, y, z); // r3Vect`

Parameters: x, y, z, w - floating point numbers

Returns: -

Description: Set new value to the vector

Example:

```
v = new r3Vect();
v.set(0.1, 0.2, 1.0);
```

Method: sub - vector subtraction

Syntax: `v3 = v.sub(v2);`

Parameters: v2 - vector to be subtracted from the given object

Returns: vector - the result of the subtraction

Description: Returns a new vector containing the result of vector operation 'v - v2'.

Example:

```
v1 = new r3Vect(1, 1, 1);  
v2 = new r3Vect(0.5, 0.5, 0.5);  
v3 = v2.sub(v1);
```

Method: fsub - vector subtraction

Syntax: v.fsub(v2);

Parameters: v2 - vector to be subtracted from the given object

Returns: -

Description: Subtracts given vector 'v2' from the vector in question.

Example:

```
v = new r3Vect(1, 1, 1);  
v2 = new r3Vect(1, 1, 1);  
v.fsub(v2); // 0, 0, 0
```

Method: dot - dot product

Syntax: f = v.dot(v2);

Parameters: v2 - vector

Returns: f - floating point number

Description: Performs dot product between 'v' and given parameter 'v2' and returns the result. The operation has no impact to the object itself

Example:

```
v1 = new r3Vect(1, 0, 0);  
v2 = new r3Vect(1, 1, 0);  
f = v1.dot(v2);
```

Method: cross - cross product

Syntax: v3 = v.cross(v2);

Parameters: v2 - vector

Returns: v3 - result vector

Description: Computes cross product 'v x v2' and returns the resulting vector. The operation doesn't impact the object itself

Example:

```
v1 = new r3Vect(1, 0, 0);  
v2 = new r3Vect(0, 1, 0);  
v3 = v1.cross(v2);
```

Method: len - lenght of vector

Syntax: length = v.len()

Parameters: -

Returns: lenght - floating point number

Description: Compute and return the lenght of the vector.

Example:

```
v = new r3Vect(0, 2, 0);  
l = v.len(); // l = 2;
```

Method: norm - normalize vector

Syntax: len = v.norm();

Parameters: -

Returns: len - original lenght of the vector

Description: Normalizes the vector and returns the original lenght.

Example:

```
v = new r3Vect(0.5, 0, 0);  
v.norm();
```

Method: orth - orthogonal projection

Syntax: v3 = v.orth(v2);

Parameters: v2 - vector

Returns: v3 - result vector

Description: Orthogonal projection. The resulting 'v3' vector is perpendicular to 'v' and on the same plane as both 'v' and 'v2'.

Example:

```
v1 = new r3Vect(1, 0, 0);  
v2 = new r3Vect(1, 1, 1);  
v3 = v1.orth(v2);
```

Method: proj - projection

Syntax: v3 = v.proj(v2);

Parameters: v2 - vector to be projected on 'v'

Returns: v3 - vector

Description: Returns vectors 'v2' projection on 'v'.

Example:

```
v1 = new r3Vect(1, 0, 0);  
v2 = new r3Vect(1, 1, 1);  
v3 = v1.proj(v2);
```

Method: neg - negate

Syntax: v2 = v.neg();

Parameters: -

Returns: v2 - vector

Description: Returns a vector pointing to opposite direction than the object in question.

Example:

```
v = new r3Vect(1, 0, 0);  
nv = v.neg();
```

Method: fneg - negative

Syntax: v.neg();

Parameters: -

Returns: -

Description: Makes the vector negative vector i.e. inverts the direction of the vector

Example:

```
v = new r3Vect(1, 0, 0);  
v.fneg();  
v.print(); // -1, 0, 0
```

Method: mul - scalar multiplication

Syntax: v2 = v.mul(f);

Parameters: f - scalar for multiplying the vector

Returns: v2 - result

Description: Multiplies the vector with the given scalar value and returns the resulting new vector.

Example:

```
v = new r3Vect(1, 2, 3);  
v2 = v.mul(10); // v2 = 10, 20, 30
```

Method: fmul - scalar multiplication

Syntax: v.fmul(f);

Parameters: f - floating point number

Returns: -

Description: Multiplies the vector by given scalar.

Example:

```
v = new r3Vect(1, 2, 3);  
v.fmul(10);  
v.print(); // v = 10, 20, 30
```

Method: div - division by scalar

Syntax: v2 = v.div(f);

Parameters: f - floating point number

Returns: v2 - vector

Description: Divides the vector by given scalar and returns a new vector holding the result.

Example:

```
v = new r3Vect(1, 0, 0);  
v2 = v.div(2.0);
```

Method: fdiv - division by scalar

Syntax: v.fdiv(f);

Parameters: f - floating point number

Returns: -

Description: Divides the vector by given scalar.

Example:

```
v = new r3Vect(1, 0, 0);  
v.fdiv(2.0);  
v.print(); // 0.5, 0, 0
```

Method: cmul

Syntax: v3 = v.cmul(v2);

Parameters: v2 - vector

Returns: v3 - vector

Description: Component wise multiplication. Multiplies each component in 'v' by corresponding component of 'v2' and returns the result as a new vector.

Example:

```
v1 = new r3Vect(1, 2, 3);  
v2 = new r3Vect(1, 2, 3);  
v3 = v1.cmul(v2); // v3 == 1, 4, 9
```

Method: fcmul - vector multiplication

Syntax: v.fcmul(v2);

Parameters: v2 - vector

Returns: -

Description: Multiplies the 'v' by given vector 'v2'.

Example:

```
v = new r3Vect(1, 2, 3);  
v2 = new r3Vect(1, 2, 3);  
v.fcmul(v2);  
v.print(); // 1, 4, 9
```

Method: cmp - compare

Syntax: b = v.cmp(v2, tolerance);

Parameters: v2 - vector
tolerance - tolerance

Returns: b - boolean, true if vectors are identical within the given tolerance

Description: Compare the given vector 'v2' against the vector in question and return true if the vectors are identical. Otherwise returns false.

Example:

```
v1 = new r3Vect(1, 1, 1);  
v2 = new r3Vect(1.1, 1, 1);  
v1.cmp(v2, 0.001);
```

Method: print - print vector**Syntax:** v.print()**Parameters:** -**Returns:** -**Description:** Print out the current value of the vector.**Example:**

```
v.set(1, 2, 3);  
v.print();
```

Method: angle - angle to given vector**Syntax:** angle = v.angle(v2);**Parameters:** v2 - vector**Returns:** angle - angle in radians**Description:** Compute the angle between the vector in question and the given parameter 'v2'.**Note**

This method is not supported by 4d vector.

Example:

```
v = new r3Vect(1, 0, 0);  
v2 = new r3Vect(0, 1, 0);  
angle = v.angle(v2);
```

Method: dist - distance to given point**Syntax:** dist = v.dist(v2);**Parameters:** v2 - vector**Returns:** dist - distance**Description:** Measure the distance between two points specified by 'v' and 'v2'**Example:**

```
v = new r3Vect(1, 0, 0);  
v2 = new r3Vect(2, 0, 0);  
distance = v.dist(v2); // distance == 1.0
```

Float Class

JavaScript does not support call-by-reference type of function parameters for basic types. Correspondingly, Realsoft 3D methods returning data back to the caller through the method parameters won't work. To overcome this limitation Realsoft 3D implements r3Float object type.

For example, the r3nurbs curve class implements method MEASURELENGTH(len), which returns the length of the curve through the function parameter 'len'. This would not be possible using JavaScript basic types:

```
// won't work
var len;
curve.MEASURELENGTH(len);
```

because JavaScript basic types are passed to function as call-by-value parameters. To allow the MEASURELENGTH() method to actually change the value of the 'len' parameter, call-by-reference is required.

```
// this works
var len = new r3Float();
curve.MEASURELENGTH(len);
```

Methods

Method: r3Float - constructor for float

Syntax: v = new r3Float()

Parameters: value - floating point number

Returns: v - new float object

Description: Creates a new float object. If you don't pass any parameters to the constructor, the value is initialized to zero.

Example: v = new r3Float(10);

Method: add - addition

Syntax: v2 = v.add(v1);

Parameters: v1 - float

Returns: v2 = result float

Description: This adds the given float to the object in question and returns the result as a new float. The operation doesn't change the value of the object.

Example:

```
v = new r3Float(0.1);
r = v.add(0.1);
```

Method: fadd - float addition

Syntax: v.fadd(v2);

Parameters: v2 - float to be added

Returns: -

Description: Float operation v = v+v2.

Example:

```
v = new r3Float(1);
v.fadd(0.1); // add 0.1 to the current value
```

Method: set - set value

Syntax: `v.set(value);`
Parameters: value - floating point number
Returns: -

Description: Set new value to the float

Example:

```
v = new r3Float();  
v.set(10.3);
```

Method: sub - float subtraction

Syntax: `v3 = v.sub(v2);`

Parameters: v2 - float to be subtracted

Returns: float - the result of the subtraction

Description: Returns a new float containing the result of float operation 'v - v2'.

Example:

```
v = new r3Float(10);  
v2 = v.sub(5);  
v2.print(); // 5
```

Method: fsub - float subtraction

Syntax: `v.fsub(v2);`

Parameters: v2 - float to be subtracted

Returns: -

Description: Subtracts given float 'v2' from the float in question.

Example:

```
v = new r3Float(1);  
v.fsub(0.5);  
v.print(); // 0.5
```

Method: neg - negate

Syntax: `v2 = v.neg();`

Parameters: -

Returns: v2 - float

Description: Returns negative value of the object.

Example:

```
v = new r3Float(1);  
v2 = v.neg();
```

Method: fneg - negative

Syntax: `v.neg();`

Parameters: -

Returns: -

Description: Makes the float negative float.

Example:

```
v = new r3Float(1);  
v.fneg();  
v.print(); // -1
```

Method: mul - multiplication

Syntax: v2 = v.mul(f);

Parameters: f - float number

Returns: v2 - result

Description: Multiplies the float with the given value and returns the value.

Example:

```
v = new r3Float(2);  
v2 = v.mul(10);  
v2.print(); // 20
```

Method: fmul - multiplication

Syntax: v.fmul(f);

Parameters: f - floating point number

Returns: -

Description: Multiplies the float by given value

Example:

```
v = new r3Float(2);  
v.fmul(10); // new value == 20
```

Method: div - division by scalar

Syntax: v2 = v.div(f);

Parameters: f - floating point number

Returns: v2 - float

Description: Divides the float by given value and returns a new float holding the result.

Example:

```
v = new r3Float(10);  
v2 = v.div(2);  
v2.print(); // 5
```

Method: fdiv - division by scalar

Syntax: v.fdiv(f);

Parameters: f - floating point number

Returns: -

Description: Divides the float by given value.

Example:

```
v = new r3Float(1);  
v.fdiv(2); // 0.5
```

Method: cmul**Syntax:** v3 = v.cmul(v2);**Parameters:** v2 - float**Returns:** v3 - float**Description:** Component wise multiplication. Multiplies each component in 'v' by corresponding component of 'v2' and returns the result as a new float.**Example:** v = new r3Vect(1, 2, 3);**Method:** cmp - compare**Syntax:** b = v.cmp(v2, tolerance);**Parameters:** v2 - float
tolerance - tolerance**Returns:** b - boolean, true if floats are identical within the given tolerance**Description:** Compare the given float 'v2' against the float in question and return true if the floats are within the given tolerance. Otherwise returns false.**Example:**

```
v = new r3Vect(1.1);  
if(v.cmp(1.2, 0.01)) {  
    print("Not identical");  
} else {  
    print("Identical");  
}
```

Method: print - print float**Syntax:** v.print()**Parameters:** -**Returns:** -**Description:** Print out the current value of the float.**Example:**

```
v.set(1.5);  
v.print();
```

Integer Class

JavaScript does not support call-by-reference type of function parameters for basic types. Correspondingly, Realsoft 3D methods returning data back to the caller through the method parameters won't work. To overcome this limitation Realsoft 3D implements r3Int object type.

Methods

Method: r3Int - constructor for int

Syntax:	<code>v = new r3Int()</code>
Parameters:	value - inting point number
Returns:	v - new int object
Description:	Creates a new int object. If you don't pass any parameters to the constructor, the value is initialized to zero.
Example:	<code>v = new r3Int(10);</code>
Method:	add - addition
Syntax:	<code>v2 = v.add(v1);</code>
Parameters:	v1 - int
Returns:	v2 = result int
Description:	This adds the given int to the object in question and returns the result as a new integer. The operation doesn't change the value of the object.
Example:	<code>v = new r3Int(0.1); r = v.add(0.1);</code>
Method:	fadd - int addition
Syntax:	<code>v.fadd(v2);</code>
Parameters:	v2 - int to be added
Returns:	-
Description:	Int operation <code>v = v+v2</code> .
Example:	<code>v = new r3Int(1); v.fadd(0.1); // add 0.1 to the current value</code>
Method:	set - set value
Syntax:	<code>v.set(value);</code>
Parameters:	value - inting point number
Returns:	-
Description:	Set new value to the int
Example:	<code>v = new r3Int(); v.set(10.3);</code>
Method:	sub - int subtraction
Syntax:	<code>v3 = v.sub(v2);</code>
Parameters:	v2 - int to be subtracted
Returns:	int - the result of the subtraction

Description: Returns a new int containing the result of int operation 'v - v2'.

Example:

```
v = new r3Int(10);  
v2 = v.sub(5);  
v2.print(); // 5
```

Method: fsub - int subtraction

Syntax: v.fsub(v2);

Parameters: v2 - int to be subtracted

Returns: -

Description: Subtracts given int 'v2' from the int in question.

Example:

```
v = new r3Int(1);  
v.fsub(0.5);  
v.print(); // 0.5
```

Method: neg - negate

Syntax: v2 = v.neg();

Parameters: -

Returns: v2 - int

Description: Returns negative value of the object.

Example:

```
v = new r3Int(1);  
v2 = v.neg();
```

Method: fneg - negative

Syntax: v.neg();

Parameters: -

Returns: -

Description: Makes the int negative integer.

Example:

```
v = new r3Int(1);  
v.fneg();  
v.print(); // -1
```

Method: mul - multiplication

Syntax: v2 = v.mul(f);

Parameters: f - int number

Returns: v2 - result

Description: Multiplies the int with the given value and returns the value.

Example:

```
v = new r3Int(2);  
v2 = v.mul(10);  
v2.print(); // 20
```

Method: fmul - multiplication

Syntax: v.fmul(f);

Parameters: f - inting point number

Returns: -

Description: Multiplies the int by given value

Example:

```
v = new r3Int(2);  
v.fmul(10); // new value == 20
```

Method: div - division by scalar

Syntax: v2 = v.div(f);

Parameters: f - inting point number

Returns: v2 - int

Description: Divides the int by given value and returns a new int holding the result.

Example:

```
v = new r3Int(10);  
v2 = v.div(2);  
v2.print(); // 5
```

Method: fdiv - division by scalar

Syntax: v.fdiv(f);

Parameters: f - inting point number

Returns: -

Description: Divides the int by given value.

Example:

```
v = new r3Int(1);  
v.fdiv(2); // 0.5
```

Method: cmul

Syntax: v3 = v.cmul(v2);

Parameters: v2 - int

Returns: v3 - int

Description: Component wise multiplication. Multiplies each component in 'v' by corresponding component of 'v2' and returns the result as a new integer.

Example: v = new r3Vect(1, 2, 3);

Method: cmp - compare

Syntax:	<code>b = v.cmp(v2, tolerance);</code>
Parameters:	<code>v2</code> - int <code>tolerance</code> - tolerance
Returns:	<code>b</code> - boolean, true if ints are identical within the given tolerance
Description:	Compare the given int 'v2' against the int in question and return true if the ints are within the given tolerance. Otherwise returns false.
Example:	<pre> v = new r3Vect(1.1); if(v.cmp(1.2, 0.01)) { print("Not identical"); } else { print("Identical"); } </pre>
Method:	<code>print</code> - print int
Syntax:	<code>v.print()</code>
Parameters:	-
Returns:	-
Description:	Print out the current value of the integer.
Example:	<pre> v.set(1.5); v.print(); </pre>

Tools

3D painting Tools

The 3D Painting tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the 3D Painting tools.



These tools can be drag&dropped to a tool bar.

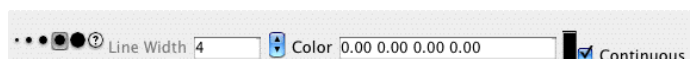
Airbrush



Paint with an airbrush on UV mapped objects.

Set tool options through the Control Bar. Press down the left mouse button on a View window over a UV mapped object and move the mouse to paint with the air brush. You can release the left mouse button and move the mouse to another place and start painting again by pressing down the left mouse button. To stop painting click Accept on the Control Bar or select Accept on the View window popup menu.

The airbrush tool has the following options:



- Line Width - Select line width for 1 to 5 pixels or custom line width.

- Line Width - Custom line width.
- Color - Color to paint.
- Continuous - If set, draws continuous line.

Circle



Paint outline or filled circles on UV mapped objects.

Set tool options through the Control Bar. Click two points on a View window over a UV mapped object to draw a circle. The first point defines the center of the circle and the second point defines a point on the circle's perimeter.

The Circle tool has the following options:



- Line Width - Select line width for 1 to 5 pixels or custom line width.
- Line Width - Custom line width.
- Filled - Outline or filled curve.
- Color - Color of the circle to paint.

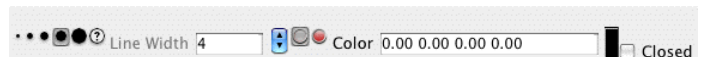
Curve



Paint outline or filled freeform curves on UV mapped objects.

Set tool options through the Control Bar. Press down the left mouse button on a View window over a UV mapped object and draw the curve while holding the left mouse button down. Release the left mouse button to accept the curve.

The Curve tool has the following options:



- Line Width - Select line width for 1 to 5 pixels or custom line width.
- Line Width - Custom line width.
- Filled - Outline or filled curve.
- Color - Color of the curve to paint.
- Closed - Closed curve.

Line



Paint a polyline on UV mapped objects.

Set tool options through the Control Bar. Click points on a View window over a UV mapped object to draw a polyline. Click Accept on the Control Bar or select Accept on the View window popup menu to accept the polyline.

The Line tool has the following options:



- Line Width - Select line width for 1 to 5 pixels or custom line width.
- Line Width - Custom line width.
- Filled - Outline or filled line.
- Color - Color of the line to paint.

Paste Colors



Paste selected objects color values on other objects' texture maps. Projects the selected object's colors on the underlying UV mapped objects.

Select the objects whose colors to paste. Click Accept on the Control Bar or select Accept on the View window popup menu.

The Paste Colors tool has the following options:

- U resolution - Subdivision of object in U direction.
- V resolution - Subdivision of object in V direction.

Picker



Color picker, read the color of the surface.

Click an object on a View window to pick its color.

The Picker tool has the following options:

- Color - Color of the picked surface point.
- Position - Surface hit point.
- Normal - Surface normal.
- UVW - UVW Coordinates.

Polygon



Paint symmetrical closed polygons on UV mapped objects.

Set tool options through the Control Bar. Click two points on a View window over a UV mapped object to draw a symmetrical polygon. The first point defines the center of the polygon and the second point defines one corner of the polygon.

The polygon tool has the following options:

- Line Width - Select line width for 1 to 5 pixels or custom line width.

- Line Width - Custom line width.
- Filled - Outline or filled polygon.
- Color - Color of the polygon to paint.
- Count - Number of points in the polygon.

Rectangle



Paint outline or filled rectangles on UV mapped objects.

Set tool options through the Control Bar. Click two points on a View window over a UV mapped object to draw a rectangle.

The Rectangle tool has the following options:



- Line Width - Select line width for 1 to 5 pixels or custom line width.
- Line Width - Custom line width.
- Filled - Outline or filled rectangle.
- Color - Color of the rectangle to paint.

Selected



Paint with the selected objects on UV mapped objects.

Select objects to be used as pens. Click Accept on the Control Bar or select Accept on the View window popup menu.

The Selected tool has the following options:



- Line Width - Select line width for 1 to 5 pixels or custom line width.
- Line Width - Custom line width.
- Filled - Outline or filled curve.
- Color - Color of the curve to paint.

UVImage



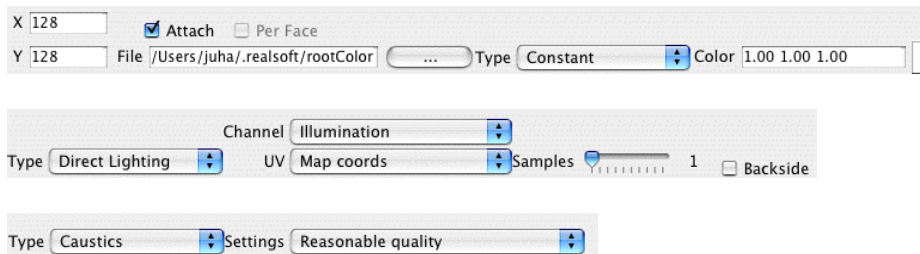
This tool creates and attaches a new image or a material with a texture map to the selected objects.

The tool can evaluate various shading components. It does so called 'baking' - procedural materials, complex illumination etc. can be evaluated and stored as a texture map. Using a texture map instead of a multi-layered procedural shader set can speed up rendering significantly. Baking is also a useful feature for scene export purposes.

To use the tool:

- Select objects to make the UV images for. For Illumination maps and surface shading, select also a light source because that helps to select the correct surface side.
- Set tool options through the Control Bar.
- Click Accept on the Control Bar or select Accept on the View window popup menu.

The UV Image tool has the following options:



- X - Pixels in X direction.
- Y - Pixels in Y direction.
- Type - Type of the image to be created.
- Constant - Initializes a blank image and assigns it to the selected channel
- Evaluate - Evaluates all materials and stores the result (in a selected channel) to a texture map
- Direct Lighting - Direct shadows and illumination from light sources
- Indirect Lighting - Global diffuse illumination bouncing from objects. Direct lighting is not included.
- Caustics - Light caustics casted by reflective surfaces
- Surface Shading - Evaluates all surface shaders from surface geometry to surface finishing. Illumination, reflections etc. are included. This option can be used e.g. for reflection map creation. The viewing direction in shading (which is relevant for reflection computation and some illumination effects) is perpendicular to the surface.
- Channel - Channel to be evaluated.
- UV - Parameter channel.
- Color - Constant fill color for the image to be created.
- Attach - Attach generated image to the selected object as texture map.
- Per Face - Attach the map as a SDS face material (for subdivision surfaces only).
- File - Name of file to write image to.
- Samples - Amount of supersamples per pixel.
- Backside - Selects the side of the surface for which illumination is evaluated.
- Settings - Defines recursion depth, threshold and traced rays.

Vertex Painter



Paint vertex properties. Select the object whose pointwise attribute you want to paint. Select the property you want to paint and the value to paint with.

Press down the left mouse button on a View window over the selected object and move the mouse to paint. You can release the left mouse button and move the mouse to another place and start painting again by pressing down the left mouse button. To stop painting click Accept on the Control Bar or select Accept on the View window popup menu.

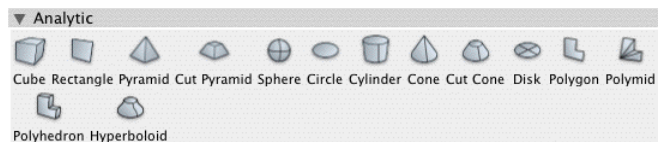
The Vertex Painter tool has the following options:



- Pen Size - Size of the circular pen.
- Pen Type
 - Solid - All parts of the pen have the same effect.
 - Gradient - The center of the pen has the maximum effect. The effect diminishes toward the perimeter of the circular pen.
- Value - Value to be painted with.
- Target - Property to be painted.

Analytic Tools

The Analytic tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains tools for creating analytic objects.



These tools can be drag&dropped to a tool bar.

Circle



Creates an analytic circle object.

Specify options in the Control Bar. Define the circle by clicking two or three points on a View window depending on the Define option. The Circle tool has the following options:



- Define - How the circle is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Filled - If checked, creates a disk, otherwise creates a circle.
- Arc - If checked creates an arc.

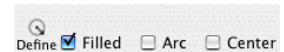
Cone



Creates an analytic cone object.

Specify options in the Control Bar. Define the bottom circle of the cone by clicking two or three points on a View window depending on the Define option. Define the apex point by clicking on a View window.

The Cone tool has the following options:



- Define - How the cone is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Filled - If not checked the top and bottom lids of the cone are not created.
- Arc - If checked creates a cone sector.
- Center - If checked, the apex will be centered over the bottom.

Cube



Creates an analytic cube object.

Specify options in the Control Bar. Define the cube by clicking two points on a View window. The Cube tool has the following options:



- Define - How the cube is defined, either by two corners or center and corner.
- Filled - If not checked the top and bottom face of the cube are not created.

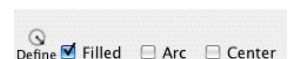
Cut Cone



Creates an analytic cut cone object.

Specify options in the Control Bar. Define the bottom circle of the cut cone by clicking two or three points on a View window depending on the Define option. Define the top circle by clicking two points on a View window.

The Cut Cone tool has the following options:



- Define - How the cut cone is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc

- Center & Corner
- Filled - If not checked the top and bottom lids of the cut cone are not created.
- Arc - If checked creates a cut cone sector.
- Center - If checked, the top will be centered over the bottom.

Cut Pyramid



Creates an analytic cut pyramid object.

Define the bottom rectangle of the cut pyramid by clicking two points on a View window. Define the top rectangle point by clicking two more points on a View window. The tool has the following options:



- Define - How the cut pyramid is defined, either by two corners or center and corner.
- Filled - If not checked the top and bottom face of the cut pyramid are not created.
- Center - If checked, the top will be centered over the bottom.

Cylinder



Creates an analytic cylinder object.

Specify options in the Control Bar. Define the cylinder by clicking two or three points on a View window depending on the Define option. The Cylinder tool has the following options:



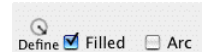
- Define - How the cylinder is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Filled - If not checked the top and bottom lids of the cylinder are not created.
- Arc - If checked creates a cylinder sector
- Center - If checked, the top will be centered over the bottom.

Disk



Creates an analytic disk object.

Specify options in the Control Bar. Define the disk by clicking two or three points on a View window depending on the Define option. The Disk tool has the following options:



- Define - How the Disk is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Filled - If checked, creates a disk, otherwise creates a circle.
- Arc - If checked creates an arc.

Hyperboloid



Creates a hyperboloid or ellipsoid segment object.

Specify options in the Control Bar. Define the bottom circle of the hyperboloid or ellipsoid segment by clicking two or three points on a View window depending on the Define option. Define the top circle by clicking two points on a View window.

The Hyperboloid tool has the following options:



- Define - How the hyperboloid is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Filled - If not checked the top and bottom lids of the hyperboloid are not created.
- Arc - If checked creates a hyperboloid sector.
- Center - If checked, the top will be centered over the bottom.

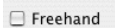
Polygon



Creates an analytic polygon object.

Specify options in the Control Bar. If the Freehand option is set, then press down the left mouse button on a View window and draw the outline of the polygon while holding the left mouse button down. Otherwise click a number of points on a View window and accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Polygon tool has the following option:



- Freehand - Define polygon by moving the mouse while holding down the left mouse button.

Polyhedron



Creates an analytic polyhedron object.

Specify options in the Control Bar. If the Freehand option is set, then press down the left mouse button on a View window and draw the outline of the polyhedron while holding the left mouse button down. Otherwise click a number of points on a View window and accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Polyhedron tool has the following options:



- Depth - Distance between of the top and the base polygon.
- Freehand - Define polygon by moving the mouse while holding down the left mouse button.

Polymid



Creates an analytic polymid object.

Specify options in the Control Bar. If the Freehand option is set, then press down the left mouse button on a View window and draw the outline of the bottom polygon while holding the left mouse button down. Otherwise click a number of points on a View window and accept the bottom polygon by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar. Define the apex point by clicking on a View window.

The Polymid tool has the following options:



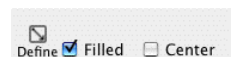
- Depth - Distance between the apex and base polygon.
- Freehand - Define polygon by moving the mouse while holding down the left mouse button.

Pyramid



Creates an analytic pyramid object.

Specify options in the Control Bar. Define the bottom rectangle of the pyramid by clicking two points on a View window. Define the apex point by clicking on a View window. The Pyramid tool has the following options:



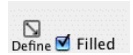
- Define - How the pyramid is defined, either by two corners or center and corner.
- Filled - If not checked the bottom face of the pyramid is not created.
- Center - If checked, the apex will be centered over the bottom.

Rectangle



Creates an analytic rectangular plane.

Specify options in the Control Bar. Define the rectangle by clicking two points on a View window. The Rectangle tool has the following options:



- Define - How the rectangle is defined, either by two corners or center and corner.
- Filled - Whether to create a surface or a border line.

Sphere



Creates an analytic sphere object.

Specify options in the Control Bar. Define the sphere by clicking two points on a View window. The Sphere tool has the following option:



- Input - How the sphere is defined, by center and radius, two corners, or center and corner.

Boolean Tools

The Boolean tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the Boolean operation tools.



These tools can be drag&dropped to a tool bar.

Boolean



The Boolean tool creates a Boolean operation object or a boundary representation solid.

Select the objects to be operated. Select the boolean operation. Select Accept from the View window's popup menu or click the Accept button on the control bar.

The Boolean tool has the following options:



- Operation
 - Union - collect one or more objects inside a new level.
 - Intersection - AND operation between the selected objects.
 - Remove - first selected object AND NOT the remaining selected objects.
 - Solid - Boundary Solid. Create new solid primitives from one or more freeform surfaces that define closed volume.
- Ignore Materials - If set, the material of the first selected object is not affected by the tool object.
- Cut Wire - If set, wireframes of boolean operands are trimmed to match the result volume.

Union



Collect one or more objects (the currently selected objects) inside a new level. The resulting object contains those parts of space that are inside any of the operated objects. The Union tool has no options.

Intersect



Apply Boolean AND operation between the selected objects. The resulting object contains those parts of space that are inside all of the operated objects.

The Intersect tool has the following options:

☒ Ignore materials ☒ Cut Wire

- Ignore Materials - If set, the material of the first selected object is not affected by the tool object.
- Cut Wire - If set, wireframes of boolean operands are trimmed to match the result volume.

Remove



Apply Boolean AND NOT operation between the selected object, i.e. the first selected object AND NOT the remaining selected objects. The resulting object contains those parts of space that are inside the first selected object but not inside any of the other selected objects. The Remove tool has the following options:

- Ignore Materials - If set, the material of the first selected object is not affected by the tool object.
- Cut Wire - If set, wireframes of boolean operands are trimmed to match the result volume.

☒ Ignore materials ☒ Cut Wire

Solid



Create a Boundary Solid. Create new solid primitives from one or more freeform surfaces that define a closed volume. The tool has no options.

Building Construction Tools

Building



This is the main tool for building construction. Other building tools described below add sub components to the manager object created using this tool. To use the tool:

- Take a top view and click the building tool button
- Define some building options (see the option list below) if necessary
- Click a sequence of points in a view window to define the outer wall for the building
- In polyline mode, click Accept to finish an open wall sequence, or close the wall loop by clicking on top of the first point

- Go to the material library of the select window. Using the property window, select some texture maps for the material templates, which the tool created.

The tool has the following options:

- Type: rectangle or polyline. Rectangle mode switches automatically to polyline drawing if the two first points define a vertical or horizontal line (a collapsed rectangle would be useless). In polyline mode, the first line can travel in any angle.
- Closing: Automatic mode draws an open line which closes automatically when the user clicks on top of the first point. Open mode keeps the line open and does not finish at overlapping points. Closed mode forces a closed wall loop, so that the tool can be finished by selecting Accept or by hitting enter key, and there is no need to aim at the first point to finish the tool.
- Auto Texture: when set, the tool adds material templates for texturing the building parts (walls, base, floor and ceiling) to the material library. The tool also adds required material maps to the geometric hierarchy of the building.
- Wall Thickness: can be defined here or changed later using the property window.

The defined wall point sequence can be found from the sub hierarchy of the building object. It is the first wall object in the 'wall definitions' sub folder. The actual point sequence is stored into a polygonal NURBS curve in the mentioned wall object hierarchy. Editing the NURBS curve updates the walls.



Note

Although the building tool accepts open wall sequences, real houses should include a closed outer wall loop in the first wall object. The same data defines directly the base and the roof edges. Splitting walls to several open sections disables automatic base and roof construction.

Hole

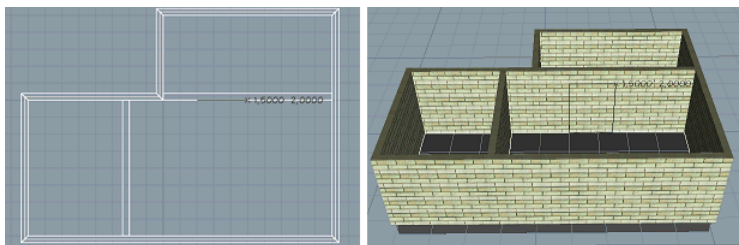


Cuts a hole to a wall. The tool has two operation modes:

- Wall surface projection
- Wall base line projection

The hole tool chooses the most appropriate projection automatically. If the hole is defined from a top view (or almost top view), drawing a rectangular hole to a vertical wall is practically impossible. Therefore, the tool lets the user to draw a line segment along the wall base lines. The mouse clicks define only the width of the hole; elevation from the floor level and height can be defined numerically or afterwards by editing the hole object.

If the modeling view shows the walls from a sufficiently perpendicular angle, the tool lets the user to draw the hole directly to the wall under the mouse pointer. All hole dimensions and the position is defined by the two mouse clicks.



Drawing a hole in orthogonal wireframe view and in perspective shaded view

To use the tool:

- Select a building. The hole tool appears to the tool bar. Click it.
- navigate a view to find a suitable modeling angle. Click two times to define the hole.

The tool adds a hole object to the sub level called 'holes' of the building object. A hole object is just a tagged rectangle object. You can move, stretch or duplicate it to get more holes. Rotated holes are not supported; hole edges must be parallel with the wall edges. Moving a hole away from the target wall turns hole-cutting feature off.

The tool does not have any toolbar options, but the dimensions and the elevation can be controlled numerically using the measuring window. The controls are useful when the hole is created from top view, because height and base elevation cannot be set with the mouse.

Roof



Adds a roof constructor object to the building. The roof tool does not have any tool options. To use it, select a building and click the roof tool. The tool creates a default roof shape immediately. The roof can be customized by editing its properties and using the Roof Line tool.

Roof Line

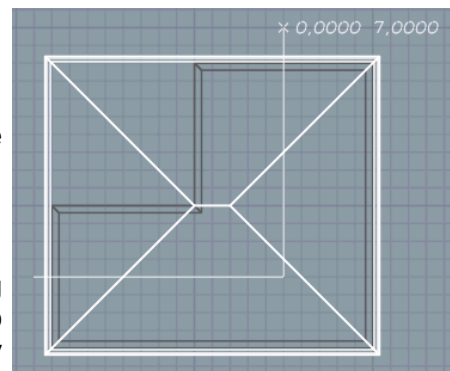


Defines the structure of the roof. The rectangular default roof can be changed to L-shaped, cross-like or to other custom shapes.

The tool is used as follows:

- Select a roof. The roof line tool appears to the tool bar. Click it.
- From a top view, click points to define a poly line. The ends of the line do not have to match exactly with the building.
- Click accept to finish the tool.

The tool can be applied multiple times to achieve complex branching roof structures. Defined roof lines are stored as NURBS objects into the 'roof lines' sub folder of the roof object. Editing the NURBS poly lines updates the roof.



Drawing a roof line



Note

The current version does not allow variable elevation of roof lines. All lines must be at the same height.

Room

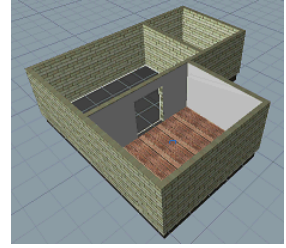


Adds a local room decoration into the building. Use the tool as follows:

- Select a building. The Room tool appears, click it.
- Set the options for the room in the tool bar. Especially the name is important, because the created material templates will refer to it. A proper name will help you to identify the respective materials.
- Click inside a room you wish to decorate
- Go to the Material library of the select window and define properties for the room materials.
- The hierarchy of the room object contains sub folders, which define mouldings. You can add an example piece of moulding to the folders.

The tool has the following options:

- Texture Walls: covers the walls with a new surface (wall paper)
- Texture Floor: covers the floor with a new surface
- Texture Ceiling: covers the ceiling with a new surface
- Add Material Templates: adds the material templates for texture mapping the created surfaces.



Note, that you have an option to add just plain white surfaces, or the surfaces and material templates for texturing them.

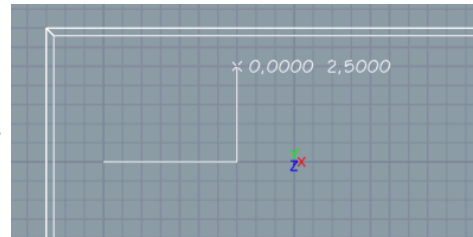
Sub Wall



Adds a new wall sequence to the building. The building can be split into rooms using this tool. Use the tool as follows:

- Select a building object. The sub wall tool appears to the toolbar. Click it to activate.
- Draw a wall point sequence. Note that when start and end snapping are active, you do not need to aim exactly at the outer wall. Just click close to the wall to which you wish to snap.
- Finish the tool by clicking Accept (or hit Enter)

The tool options are the same as for the building tool above with some exceptions. Auto texture option is not available; inner walls can be textured using the room tool. Snap Start/End options will search for the nearest walls and match the ends of the new wall respectively.



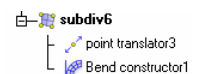
Sub wall will auto snap to outer walls

Construction Tools

Introduction

All tools in the construction tool group create an object, which applies a certain effect to its target objects. Any number of construction objects can be created to control an object.

For example, a *point translator* object can deform some of the points in a SDS object. Bend deformer can then be used for bending the deformed object. In this case, there would be two deformations associated with the SDS object.



A subdivision object controlled by point translator and bend objects



Bend deforms a lattice which deforms a SDS object

Construction objects can also be applied to other construction objects. For example, a *lattice deformer* can be used for bending a SDS sphere. A *bend deformer* can be used for bending the lattice deformer. This leads to a hierarchical construction system, where any object can have any number of constructors.

A certain construction effect can be modified by editing the attributes of the corresponding constructor object. To remove an effect of a constructor, delete the constructor object.

Construction based tools can be found from the Construction tool group.

Construction Tools

Bend Deformator

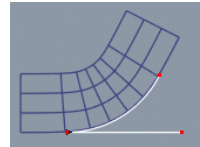


Bend deformator is a constructor based deformer. It can be used for generating circular bend effects to free form objects.

To bend a nurbs mesh:

1. Select the target object of bending and activate the Bend Constructor tool. Then define bending axis by clicking two times in a view window.

2. Switch the created bend object to point edit mode and drag the end point handle to bend the axis. The target object bends with the axis.



A bend deformator applied to a nurbs mesh

Carriage Tool

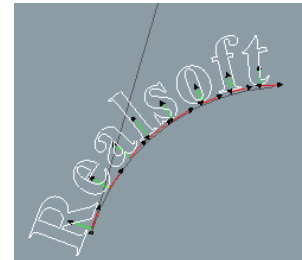


This tool creates so called 'carriage' object for the currently selected target objects. The object consists of a single coordinate system handle, through which the target objects can be transformed.

To use the tool:

- Select desired target object(s).
- Activate the tool and enter two points through the view window to define the coordinate system for the carriage object.
- Modify the carriage object and the target objects are modified accordingly.

The following image demonstrates the effect of the object.



Bend applied to a font curves through a set of carriage objects. Thanks to carriage objects, each font object is only translated and rotated, not bent

Construction Bind Tool



This tool can be used for binding constructor objects, such as Skeleton, Radial Deformator, Point Displacer and 3D lattice to their target objects. The tool can be used when you have to bind only some of the objects in a certain level.

To use the tool, select one or more target objects and then the constructor object to be bound, in this order. Activate the tool and click 'Accept'.

The tool shows you also one option 'Recursive'. If this option is set, the tool will bind also the sub objects of the selected target objects. Otherwise only the selected objects will be bound.

Displacer



Displacer is a very powerful constructor based deformer object. It allows the user to deform geometric objects using material mapping objects. It can be used, for example, to create waving flags and landscapes, just to name a few examples.

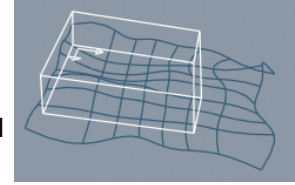


A displaced mesh consists of a mesh and a displacer object

The displacement effect is defined by a material mapping object, which is given as a sub object for the displacer.

To create a waving flag:

1. Create a nurbs mesh consisting of 10 x 6 control points and representing a flag.
2. In the material selection tab, select the desired material and create a parallel mapping object.
3. Multi select the nurbs mesh and the material mapping object, and click the displacer tool.
4. In the property window, set the 'Space' option to 'UVW'.
5. Create a key frame animation where the parallel mapping object moves over the mesh.



A waving flag generated by moving a displacement deformer over a mesh

Drag Tool



To use the tool:

- Create an animated object (say, a spinning wheel or a walking character)
- Create an object representing the ground, say an analytic rectangle.
- Multi select both objects: first animated object and then the ground object.
- Click the tool and enter two points in the view window to define direction and strength of gravity. Use the property window to set desired rebound energy and surface friction for the drag object.

Inverse Kinematics Tool



This tool creates so called Inverse Kinematics object for the currently selected skeleton object. If any of the joints in the selected skeleton is inside the created I.K object, the I.K object is automatically bound to the joint. Because binding is based on inside/outside test rather than to selected joints, is easy to save animated I.K objects to a file and use them later to animate other skeletons.

Moving the I.K object will apply Inverse Kinematics to the attached joint.

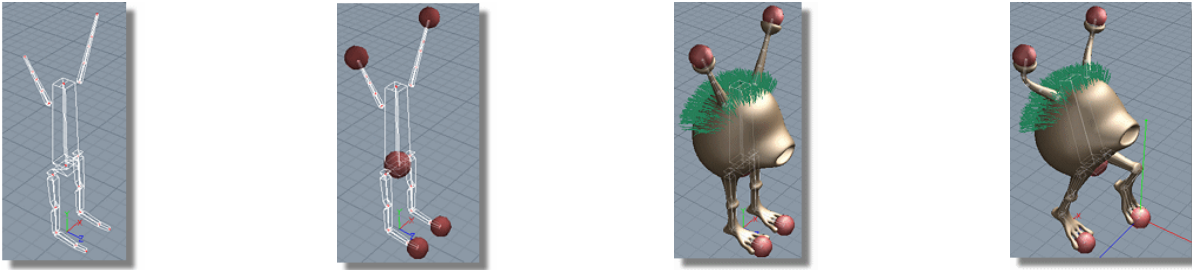
Example 1:

- Create several I.K objects and use motion capture system to define their translations.
- Load in desired skeleton and match desired skeleton joints to desired I.K objects.
- Multi select all I.K objects and click 'Bind'.
- Bind desired body to the skeleton.

Example 2:

- Create a skeleton and use move tool to animate its position (transform it in the object mode so that you have a rigid moving skeleton).
- Set Anchor for the root joints of the both legs.
- Use the tool to create I.K objects for the feet. Modify the positions and radiuses of the I.K objects so that they match the tail joints of the legs.
- Multi select all the I.K objects and click 'Bind'.

- Animate the I.K objects according to the movement of the skeleton to create a walking object.

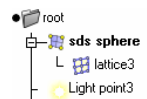


Attach I.K objects to a skeleton. Attach a body to a skeleton. Animate I.K objects.

Lattice



Lattice object is a global deformer object. Create it into an object and all the geometric points in the object are influenced by the lattice object.

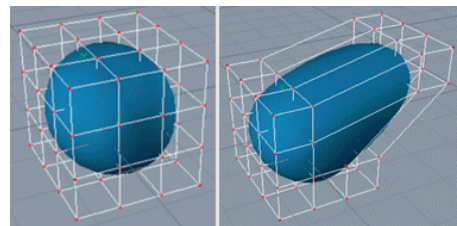


The lattice object can be used as a deformation tool for modeling purposes, or as an animation tool. When activated, the tool provides three integer fields which define the point counts in three dimensions. *Lattice applied to a SDS sphere*



Number of points in x, y and z dimensions

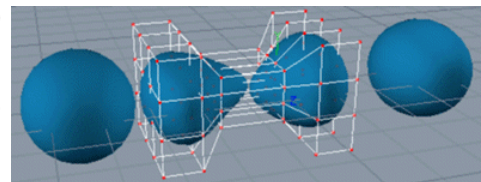
One can create a lattice over any object and deform it to deform the target object.



A SDS sphere deformed by a lattice object

You can also move an object through a deformed lattice object to get a deformation effect applied.

Lattice object properties are explained in models/geometrics/lattice documentation.



A SDS sphere moving through a lattice

Look At Tool



This tool creates a 'lookat' object. The object rotates the target objects about their origins so that their relative orientations with respect to the lookat object is maintained. For example, you can create a large number of planar trees, and use this object to control the orientation of the trees so that they will always face towards the camera.

The lookat object also defines a direction vector, which controls the banking of the target objects.

To use the tool.

- Create an object representing a gun.
- Select the gun object and create a lookat object so that the gun will point directly towards the lookat object.
- Move the lookat object sideways. The gun will be rotated accordingly.

The lookat object defines two options for controlling the rotation:

- Pitching On/Off.
- Banking On/Off

These can be set in the property window.

Move



Move constructor translates the target objects based on the specified translation vector.

To create a move constructor:

1. Select the desired target objects and activate the Move constructor tool.
2. Enter two points through the view window to define translation vector.

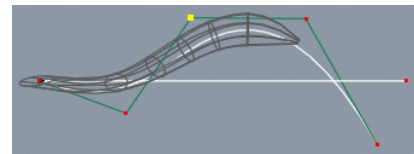
Path Level



Path Level object is similar to bend deformer in that it can be used for bending geometric objects. A path level defines a deformation using a nurbs curve.

To create a wriggling snake:

1. Create a horizontal tube representing a snake.
2. Animate the snake to move horizontally.
3. Activate the Path Level tool and define an initial bending axis along the tube. Then single point edit the nurbs curve found inside the Path Level object to get desired deformation effect. Play the animation to see a wriggling snake.



A suitable banking option can be selected through the property window.

Radial Deformator



Radial deformer is a constructor based deformer. The shorter the distance to the center point of the deformer the stronger the deformation effect.

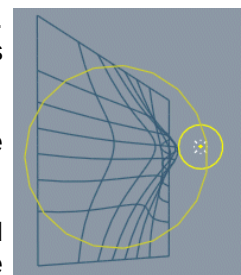
The initial position and radius of the deformer represents the state of no deformation. The desired deformation effect can then be achieved by editing the center and the radius of the sphere.

All points inside the initial radius will be influenced by the deformer. All points outside the sphere will not be affected by the deformer.

By moving the center point, one can achieve a magnet like effect. All points are moved with the sphere. The higher the distance of the point and the center of the initial sphere the smaller the translation.

To deform a nurbs mesh:

1. Create a nurbs mesh consisting of say 10 x 6 control points
2. Activate the radial deformer tool and specify two points defining the center point and radius attributes for the deformer.



A radial deformer applied to a nurbs mesh

3. Switch to edit mode and drag the center point or the radius handle of the deformer. The target points are deformed accordingly.

Setting the radius attribute to a smaller value generates a gravity like effect. In other words, target points are scaled towards the center point rather than translated with it.

The strength factor can also be controlled by defining a deformation curve through the property window. The default curve is linear. The beginning of the curve represents the maximum distance of influence and the end point corresponds to the zero distance.

The deformer also supports fractal noise deformation effect. The strength and density of the noise effect can be defined through the property window. This option is useful for creating turbulence effects. For example, a star ship can be moved through a noise field.

Point Translator



Point Translator can refer to one or more geometric points and define point wise translations for the points. In this respect, it is similar to the *Sub Group* object. However, the point translator is based on the construction system whereas the sub group is based on the lattice mapping system.

Point translators are typically used with skeletons to define facial expressions and other muscle deformations.

For example, one can create a point translator for the points representing a nose and animate the point translator to get facial expressions defined for the nose.

To animate a nose of a human head with the point translator:

1. In edit mode, select the points representing the nose of a head
2. Click the Point Translator tool. A point translator object is created.
3. Activate the 'Construct' option for the point translator.
4. Modify the point translator object in animation recording mode to animate the nose points.

Properties defined by the point translator object are explained in models/geometrics/pointrtranslator page.

Rotate



Rotate constructor rotates the target objects about the specified axis.

To rotate the selected objects by three full circles:

1. Click the 'Rotate' constructor tool to activate it:
2. Enter two points in the view window to define rotation axis.
3. Specify the desired rotation angle through the property window. To make the target object to rotate three full cycles, enter $3 * 360$.

Skeleton



Skeleton is a special purpose deformation object typically used for controlling bodies in character animations. When the skeleton is modified, the attached objects are modified accordingly.

To create a skeleton representing a leg:

1. Activate the Skeleton tool from the 'Construction' tool tab.
2. Take a side view. Click the first point at the hip bone. Now, move the mouse towards the knee but don't fix the position for the knee yet. We will have to define proper constraints for the thighbone first. Rotate the mouse upwards so that it represents the knee in its uppermost position. Then move the mouse to the opposite direction so that it shows the position of the knee in its back position. Then move the mouse to a position between the two extreme angles and to a suitable distance (the length of a thighbone) and fix the second point (the knee) by clicking the mouse.
3. Then move the mouse towards the ankle and repeat the steps above in order to define constraints for the shinbone as well.
4. Next define a bone from the ankle to the tip of the toes (we do not create toe bones this time) with appropriate constraints. Then select 'Accept' from the pop-up menu to terminate the skeleton creation.

Then you can bind the skeleton to a target object (such as a SDS mesh representing a human body) using appropriate binding tools.

Skeletons can also be grouped to hierarchical structures to represent, for example, the bone structure of a human body.

To create a hierarchical skeleton, say a spine with two legs:

1. Switch to the side view and create a skeleton representing the spine of a human body.
2. Create a skeleton representing a leg.
3. Drag & drop the leg inside the spine. The leg is now a sub object of the spine.
4. Repeat the steps 3 and 4 to create another leg. Then select the legs and click the 'Attach' tool. This attaches the legs to the spine so that if you drag one of the leg joints far enough, the effect will propagate to all three skeletons.



Binding Tools

When a skeleton and one of its target objects are selected the control bar shows the following skeleton map specific tools:

Bind

In edit mode, this tool binds the currently selected target points to their nearest currently selected joints. In Object mode all points in the selected target objects are bound to their nearest bones of the selected skeleton.

To use the tool, multi select one or more target objects and a skeleton object. Then switch to point edit mode and select the desired bones and the target points. Click the tool and the selected points are bound to the nearest selected joints.



Note

In case more than one joint is selected, each point gets mapped only to the nearest one! To bind a point to the nearest bone, just select the point and all the joints in the skeleton.

Unbind

This tool unbinds the selected points from the selected joints. If the target object is hierarchical, its sub objects are not bound.

To use the tool, multi select a target object and a skeleton object. Then switch to edit mode and select some joints and target points. Click the tool and the selected points are unbound from the selected joints.

Let's assume some of the points representing a nose got bound to a wrong skeleton. To unbind the points from the skeleton:

- Multi select the points to be unmapped and all the joints in the skeleton from which the points should be unbound. The tool is now shown in the control bar.
- Click the tool.

To unbind a point that got bound to a wrong joint:

- Select all points in the target object
- Select the joint in question
- Apply the tool

Points associated with the selected joint become unbound.

Select

The tool selects all target points bound to the currently selected joints.

The tool is only shown when both the target object and the skeleton object is selected.

The tool is useful, for example, when you want to view the points bound to a certain bone. Or, when some of the points got bound to a wrong bone and you want to unbind them. Just select the bone and click the 'Select' tool to select the target points as well.

Deselect

Deselect all the target points bound to the currently selected joints.

The tool is only shown when two objects are multi selected: a skeleton object and its target object. The tool is useful when binding points manually to complex hierarchical skeletons. It allows the user to exclude points already mapped to other skeletons.

Fidelity

This tool can be used for setting point wise fidelity.

To use the tool, select the desired target points and the associated joint. Then specify desired fidelity value through the fidelity field.

Fidelity is a special form of multi binding, based on the fact that in most cases multi binding operates over subsequent joints so that points near a joint behave like an elastic skin.

If the fidelity for a point is 0, then the point is bound only to one joint. If the fidelity is 0.5, then the point is bound to its closest joint with a weight 0.5 and to the next joint with a weight 0.5. In other words, such a point is influenced by two joints.

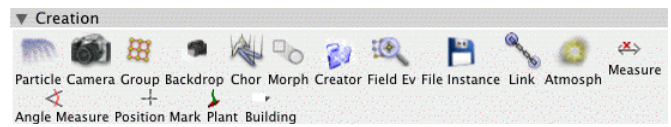
If the fidelity is 1, then only the next joint controls the point.

Weight

Sets a multi binding weight for the selected points. Select the target points and the associated joints. Then specify a weight value through the weight field.

Creation Tools

The Creation tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains some creation tools.



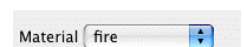
These tools can be drag&dropped to a tool bar.

Atmosphere



Creates an Atmosphere object, which defines volumetric properties for the entire space.

The atmosphere tool has the following options:



- Material - Material to be used for defining the atmospheric effects. Only VSL materials that contain the Volume phase can be used.

See also: Atmosphere Object.

Backdrop



Creates a backdrop object for the currently selected camera. If no camera is selected then a camera is created as well.

The tool has the following options:



- Distance - distance between the image and the camera.
- Material - name of the material to be mapped to the backdrop image.
- New Image - create new texture map.
- Color - create constant color backdrop.

See also: Backdrop Object.

Camera



You create a camera object by defining three points on a View window. The first point determines the origin of the camera. The second point determines the aiming direction. The third point determines the viewing angle (and the focal length) of the camera lens.

See also: Camera Object.

Choreography



This tool creates a choreography object for hierarchical time lines.

Let's imagine you have a large number of animated objects. Correspondingly, the choreographer window shows you a large number of time lines. In order to control the timing of all the choreographs through a single time line:

1. Click the tool to create a choreographer object.
2. Move all the animated objects inside the newly created choreography object.
3. Make sure the choreography object is selected and open the Choreographer window. Now you can see only one time line through which you can control all the sub time lines.

See also: Choreography.

Constraint



Create a constraint object for the selected objects. When any of these target objects are modified, the constructor object's script, as defined in the Spec tab of the property window, will be executed. Constraint objects can be used to set up relations between geometric objects, e.g. to move an object to a position that is defined by the sub object of the constraint.

The constraint tool has the following options:

- Recursive - if set, the constraint object is bound also to the sub objects of the selected target objects.

The constraint tool creates a new choreography to the selected objects. The position in the choreography stack determines when the constructor object's script is executed.

The constructor object defines the following variables:

- Target - target object
- Self - the constructor object the script is associated with
- Time - current time
- P - the current target attribute (associated with the choreography). The type of the variable depends on the the type of the target attribute (vector, vector4, float, integer, axis, coordinate system).
- Type - type of the current target attribute
- Tag - attribute identifier of the current target attribute
- Index - attribute index of the current target attribute, or -1 for regular attributes

Example:

- select a set of objects to be controlled
- create a constraint object
- open the property window and enter desired script through the Spect tab.

See also: Constraint Object.

File Instance



This tool creates an object which instances specified Realsoft 3D project file (.r3d).

The object is visualized as a bounding box, representing the actual bounding box of the referred project.

The created file instance object can be transformed similar to any geometric object.

Because the actual data resides on a disk, File Instance objects can be used for constructing complex scenes.

Only geometric objects, materials and particle effects are loaded from the specified project.

To use the tool to instance a project:

1. Click the tool to activate it.
2. Select the desired .r3d file through the shown tool control bar.
3. Click the Accept button.

Instead of accepting the tool by clicking the Accept button, you can enter two points through the view window to define bounding box for the project.

The property window allows the user to disable any of the three chunks.

Group



The Group tool is used to create objects that contain "selection sets" of NURBS mesh, NURBS curve, or Subdivision object points. First select the points, then activate the Group tool and accept.

See also: Group object.

Instance



Creates instances of the selected objects. An instance duplicates the geometry of the target object with a private matrix transformation.

Link



The link tool creates link objects that point to the currently selected geometric objects. A link object can be used where a geometric object would be evaluated.

See also: Link Object.

Metaball



Create a Metaball object.

Select one or more analytic spheres and then Accept to create a Metaball object.

Morphing



This tool creates a morphing object. The object interpolates its sub objects and assigns the result to its target objects.

To create a morphing object:

1. Activate the tool

1. Multi select desired key objects. The selected objects must have identical hierarchical structure and vertice count.

3. Accept the tool.

Morphing is set up. Play the animation to see how the target object interpolates through the specified key objects.

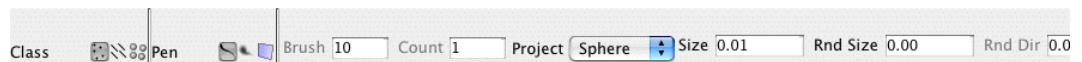
See also: Morphing.

Particle



Create clouds of 1D, 2D, or 3D particles by painting with the mouse.

The particle tool has the following options:



- Class - Particle class, possible values are
 - 1D Particle - is defined by a single point
 - 2D Particle - consists of two points. So, it has properties such as 'direction' and 'length'.
 - 3D Particle - is represented as a coordinate system
- Pen - Pen type, possible values are
 - Pencil - creates one particle for each mouse movement
 - Brush - The number of particles created for each mouse movement can be defined through the Count field.
 - Object - Emit particles from a selected curve. This creates particles along the selected NURBS curve. The number of particles generated for each mouse move event can be defined using the Count control.
- Brush - controls the radius of the airbrush particle tool. The size is measured in pixels.
- Count - Defines the number of particles created per each mouse movement when using the airbrush and selected object pens.
- Project - controls the depth coordinate of particles added by the airbrush pen. Three choices are available:
 - Disc: Particles are created inside a circular area defined by the brush in the input plane.
 - Sphere: Each mouse movement adds a new spherical cloud of particles around the mouse position defined by the input plane. The diameter of the particle cloud is defined by the Brush size control.
 - Surface: Particles are added to the surface of the selected objects. The surface also defines the orientation of the particles.
- Size - controls the size of created particles. With 2D and 3D particles, the size controls the distance of the actual 3D points defining each particle. With 1D particles, the size is stored as a pointwise rendering attribute.
- Rnd Size - randomize the particle size. Value 0 makes all particles equally big (as defined by Size above). Value 1 picks a totally random value between 0 and the value of the Size field.

- Rnd Dir - randomize particle direction. Not available for 1D particles, which do not have any direction. Randomization is controlled with 2 values. The first value defines how much the axis of each particle changes randomly. In case of 3D particles, the controlled axis is the z axis. The second value (unused for 2d particles which are symmetric) controls the random rotation around the axis.

The greater the values, the more randomized direction each particle gets. For the first component, value 1.0 already randomizes the direction 180 degrees, but the original direction still dominates the average of directions. For the second random component, value 1.0 gives a totally random rotation around the axis.

For example, when brushing tree leaves onto a ground surface, the first random component can be zero, because the flat leaves lie along the ground. The second component should be one, so that leaves have a totally random direction on the ground. In other words, a suitable random value pair is "0 1".

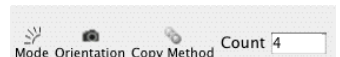
See also: Particle Objects.

Rotator



Create a rotator object for rotating copies of selected objects about a given point or aligning them along a given line. Select the object(s), activate the Rotator tool and set options. Define a point (Rotate mode) or a line segment (Align mode) on a View window.

The rotator tool has the following options:

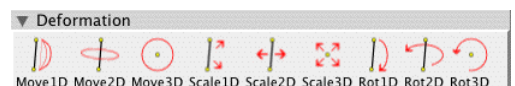


- Mode - Defines whether to rotate or align.
- Orientation - Defines how rotation axis is given. This control is available only when Rotate mode is selected.
 - Front
 - Side
- Copy Method - Defines whether to duplicate or instance the selected object
 - Instance
 - Duplicate
- Count - Number of copies

See also: Rotator Object.

Deformation Tools

The Deformation tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the deformation tools.



These tools can be drag&dropped to a tool bar.

Move1D



Move with deformation as a function of projection to an axis.

Define the axis by clicking two points on a View window. Each point is projected to the axis to get a deformation parameter. Points that project to near the first end of the axis get smaller deformation parameters than points that

project near the second end. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click on a View window to start the deformation. Move the mouse to deform and click to accept the tool.

Move2D



Move with deformation as a function of distance to an axis.

Define the axis by clicking two points on a View window. Distance to the axis is calculated for each point to get a deformation parameter. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click on a View window to start the deformation. Move the mouse to deform and click to accept the tool.

The Move2D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry
 - LatticeCoords
 - Handles
 - Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.

Move3D



Move with deformation as a function of distance to a point.

Click two points on a View window to define a circle. The radius of the circle defines how far the deformation is effective. Distance to the center of the circle is calculated for each point to get a deformation parameter. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click on a View window to start the deformation. Move the mouse to deform and click to accept the tool.

The Move3D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry

- LatticeCoords
- Handles
- Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.

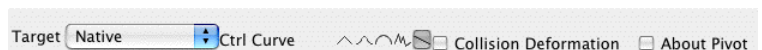
Rotate1D



Rotate with deformation as a function of projection to an axis.

Define the axis by clicking two points on a View window. Each point is projected to the axis to get a deformation parameter. Points that project to near the first end of the axis get smaller deformation parameters than points that project near the second end. The control curve is then evaluated by the deformation parameter to get the amount of deformation. click on a View Window to define the center of rotation. Click on a View Window to define the point you want to rotate about the rotation center. Move the mouse until desired rotation is achieved and click to accept the tool.

The Rotate1D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry
 - LatticeCoords
 - Handles
 - Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.

- About Pivot - Rotate object about the pivot point.

Rotate2D



Rotate with deformation as a function of distance to an axis.

Define the axis by clicking two points on a View window. Distance to the axis is calculated for each point to get a deformation parameter. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click on a View Window to define the point you want to rotate about the rotation center. Move the mouse until desired rotation is achieved and click to accept the tool.

The Rotate2D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry
 - LatticeCoords
 - Handles
 - Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.
- About Pivot - Rotate object about the pivot point.

Rotate3D



Rotate with deformation as a function of distance to a point.

Click two points on a View window to define a circle. The radius of the circle defines how far the deformation is effective. Distance to the center of the circle is calculated for each point to get a deformation parameter. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click on a View Window to define the point you want to rotate about the rotation center. Move the mouse until desired rotation is achieved and click to accept the tool.

The Rotate3D tool has the following options:



- Target - Transformation target.

- Native
- Geometry
- LatticeCoords
- Handles
- Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.
- About Pivot - Rotate object about the pivot point.

Scale1D



Scale with deformation as a function of projection to an axis.

Define the axis by clicking two points on a View window. Each point is projected to the axis to get a deformation parameter. Points that project to near the first end of the axis get smaller deformation parameters than points that project near the second end. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click two points on a View window to define the scaling center and scale. Move the mouse to deform and click to accept the tool.

The Scale1D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry
 - LatticeCoords
 - Handles
 - Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle

- Random
- Linear
- Collision Deformation - Activate collision detection system.
- About Pivot - Scale object about the pivot point.

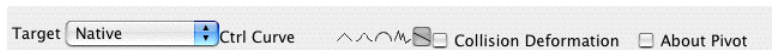
Scale2D



Scale with deformation as a function of distance to an axis.

Define the axis by clicking two points on a View window. Distance to the axis is calculated for each point to get a deformation parameter. The control curve is then evaluated by the deformation parameter to get the amount of deformation. Click two points on a View window to define the scaling center and scale. Move the mouse to deform and click to accept the tool.

The Scale2D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry
 - LatticeCoords
 - Handles
 - Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.
- About Pivot - Scale object about the pivot point.

Scale3D



Scale with deformation as a function of distance to a point.

Click two points on a View window to define a circle. The radius of the circle defines how far the deformation is effective. Distance to the center of the circle is calculated for each point to get a deformation parameter. The control

curve is then evaluated by the deformation parameter to get the amount of deformation. Click two points on a View window to define the scaling center and scale. Move the mouse to deform and click to accept the tool.

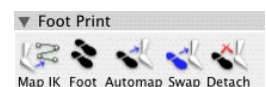
The Scale3D tool has the following options:



- Target - Transformation target.
 - Native
 - Geometry
 - LatticeCoords
 - Handles
 - Texture
- Ctrl Curve - Shape of deformation parametrization
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- Collision Deformation - Activate collision detection system.
- About Pivot - Scale object about the pivot point.

Foot Print Tools

The Foot Print tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the foot print tools.



These tools can be drag&dropped to a tool bar.

Automap



Map hierarchical skeleton to foot print object.

Select one or more skeleton joints and one I.K. object and then accept to tool.

Detach



Detach skeleton from a foot print object.

Select the skeleton and the foot print object and accept the tool.

Foot



Create foot prints defining an I.K. path.

Enter points for the foot prints and accept when done.

Map IK



Map skeleton to I.K. Path.

Select one or more skeleton joints and one I.K. object. Select Accept to map skeletons to the I.K. object.

Swap



Attach skeletons to a new foot print object.

Select the skeleton and the new foot print object and then accept the tool.

Geometry Manager Tools

3D Measure



Creates a 3D measure object, which measures surface area and volume of its sub objects.

To use the tool:

- Select objects to be measured
- Click the tool button

You can also create the measure object first without object selection and later drop objects into its hierarchy.

See also: 3D Measure Object.

Angle Measure



Creates an Angle Measure object, which displays the angle between its two control lines.

To use the tool:

- Click the tool button
- Click in a view window to define the origin of angle
- Click second time to define the first control line
- Click third time to define the second control line. This finishes the tool.

See also: Angle Measure Object.

Measure



Creates a Measure object, which displays the distance between two points.

To use the tool:

- Click the tool button
- Click in a view window to define the start point
- Click second time to define the end point. This finishes the tool.

See also: Measure Object.

Packer



Packer Tool creates a geometry manager object for packing and aligning objects. The tool has one control option: Orientation. When Orientation is set to Detect, the tool finds out the axis along which the pivot points of the selected objects have been aligned, and packs the selected objects to that axis. The user can also explicitly select one of the packing directions X, Y and Z.

After the tool is started, the user can define the direction and size of the packer box with two mouse clicks. The first click position defines the corner, which does not move when packer dimensions are later changed. If the tool is accepted before the box is drawn, the packing directions are taken from the first selected object.

Position Mark



Creates a Position Mark object, which displays the 3d coordinates of the control point.

To use the tool:

- Click the tool button
- Click in a view window to define the position. This finishes the tool.

See also: Position Mark Object.

Replicator



The tool creates a replicator object, which automatically repeats its sub objects to fill its control box. An example: select a kitchen cabinet and create a replicator. Stretch the replicator to get an entire row of kitchen cabinets.

The Orientation option of the tool works the same way as in the Packer tool above.

View Control Tools

This tool set provides the tools for controlling various View Window specific functions. The tools can be drag&dropped to a tool bar.

Persp

-  Sets perspective projection for the view on/off. The parallel camera projection is often more suitable for accurate modeling purposes.

Front

-  This tool puts the view camera to the front view orientation. The camera will look against z axis (heading 180 degrees, pitching zero). The camera is rotated to the front view about its aimpoint. Also the input plane is rotated so that its normal matches the z axis. Shift clicking the icon gives the opposite view (the back view).

Side

-  The side view tool. Side view is the view from the left side of the object (along positive x axis). The side menu is similar to the front view above. Shift clicking the icon gives the opposite view (the right side view).

Top

-  The top view tool. View camera and input plane are rotated so that the camera looks down against y axis. See the front view tool above for the side menu explanations. Shift clicking the icon gives the opposite view (the bottom view).

Reset

-  Resets the view back to the 'Native state' that has been stored into it. This usually changes camera and input plane positions and orientation, as well as the zoom scale.

Fit

-  The autofocus tool. The view camera is aimed at the center of the selected objects and zoomed/moved so that the selected objects fill about 75% of the view window. Also the clipping range is adjusted so that all selected objects become visible.

Fetch

-  Fetches the view camera orientation from the selected object. The view will be moved to its position and rotated to match its object space. For example, you can take the view from a camera object or from a spot light this way.

Assign

-  Assign view's orientation to the selected objects. Usually this tool is used for recording a view to a camera object, but you can also set spot light orientation this way, for example.

Grid

-  Enables and disables grid drawing for the currently selected grid. To select a grid, drag it from the select window's grid tab and drop it into the view window.

Snap

-  Enables and disables grid snapping. A grid must be also assigned for the view, otherwise snapping does not work. You can assign the grid by drag and dropping it from the select window into the view window, or using the view property window.

Track

-  Set view window to camera tracking mode. The view window will track the current camera during animation playback.

Render

-  Render the current view using the photo realistic renderer. The View is rendered according to the current rendering settings. You can set rendering settings for a view window by dragging and dropping the desired rendering settings from the select window into the view window, or by using the view property window.

Postproc

-  Post process the the current view window. The tool loads the previously rendered image and applies the post effects to it.

This tool works only if the view window has 'Backup Ray Trace' option set.

Shaded

-  A tool for managing OpenGL rendering. Allows the user to toggle between shaded and wireframe opengl modes. If the view window uses some other than OpenGL rendering, then the tool appears in conflict state and clicking it will change the View's rendering mode to OpenGL.

IP by 3P

-  Reads three points and fits the input plane to match the defined plane. The line between the two first points will become the input plane's x axis. For example, you can fit the input plane to any triangle or rectangle by applying this tool: activate the tool and then LMB drag three corner points with the depth modifier key ',' (comma) held down.

IP by N

-  Fits the input plane to surface tangent. Activate the tool and LMB click over a surface. You may also LMB drag (preferably grid drawing on) to see how the input plane rotates; releasing the mouse button sets the input plane.

Boxren

-  Define Render Box and enable box rendering. When enabled, the user can LMB drag a rectangle whose contents will be rendered. All later renderings will be limited to the defined area until the render box is deactivated.

Zoom

 This is a combined pan and zoom tool. To use it: Activate the tool. Then LMB drag a rectangle on a view window. Note how the dragged rectangle has automatically the proportions of the view window. After left-to-right dragging, the defined rectangle area is magnified to fill the entire view. Reversed dragging from right to left will shrink the current view area to fit the defined rectangle.

To just pan, without zooming, just click the mouse button once, without dragging.

Walk Thru

 Walk Thru animation tool.

The tool allows you to navigate in a scene. The navigation can be recorded and used in the following ways:

- Create a camera out of the recorded data
- To attach the recorded motion to the selected objects

The tool uses the mouse and keyboard input to control the orientation of the camera similar to many games: moving the mouse controls the direction and the four keys a, w, d and s allow you to move forward, backward and sideways. Collision detection system is used for automatic floor and wall detection.

To use the tool, press down the mouse in a view window. This will activate the navigation for the window and you can then control the orientation of the camera by moving the mouse.

The keys 'w', 'a', 'd' and 's' can be used for walking forward, left, right and backwards. The mouse wheel controls the focal length (zoom) of the camera.

The tool supports large number of options, well known from real world cameras.

Options

Action	Controls whether the tool creates an animated camera or animates the selected objects.
Walk	Set the maximum walking speed (m/s). The Shift key can be used for slowing the velocity down. The actual speed can also be slower when approaching walls: the tool automatically slows the user down near walls to give the user more time to react.
Turn	Controls the rotation speed.
Height	The height of the Observer (altitude). The camera will be elevated this much above the ground.
Threshold	Controls the key generation in motion recording. This value is relative to the defined turning and walking speed. The value of 0 means the motion is always recorded, no matter how much the camera moves or turns. This will result lots of key frames. The value of 1.0 means the motion is practically recorded only when the camera rotates full 360 degrees, or moves more than the given walking speed.
Gravity	Gravity on/off. If gravity is off, the user will be able to climb on the walls and fly.
Collision	Enable or disable Collision Detection. If the option is set, then the tool will detect floor and walls and it is not possible to pass through any surfaces.
Stabilizer	Image stabilizer. Smooths out shaking and sudden jumps.

Auto Focus	Auto Focus. Sets focusing distance automatically by detecting the distance to the nearest surface in front. When enabled, the tool renders a focusing frame representing the users dimensions on the scene. Autofocus option is useful when rendering animations with Dept-of-Field effect.
Input Plane	Manage Input Plane orientation. The orientation for the input plane is determined automatically from the surface in front of the camera. If the Input Grid is not shown, then the tool renders a small rectangle to visualize the input plane. This option is useful when modeling new objects to the scene: just walk thru the scene and create objects while you go.
Record	Enable camera motion recording. When set all navigation actions are recorded.
Play	Preview the recorded animation data.

Quad

 The default user interface in Realsoft 3D supports single and quadview modeling environments. This tool can be used for toggling between the two.

Lattice Mapping Tools

Lattice mapping system is based on the idea that all geometric objects define three dimensional parameter space (also called uvw space). For example, the 'u' coordinate for the nurbs curve runs along the curve whereas the two other dimensions are perpendicular to the curve.

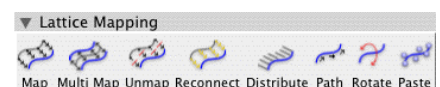
The parameter space for the nurbs mesh is such that the first two dimensions corresponds to the parameter space of the nurbs mesh whereas the third dimension corresponds to the surface normal of the mesh.

Any object can be mapped into this parameter space, in which case the coordinates of the mapped object are defined in the parameter space of the lattice object rather than in the world space. When the lattice object is then modified, the mapped geometry will get modified accordingly.

Lattice mapped objects can be animated. For example, a sphere moving in 'x' direction in the parameter space of the curve means that the sphere moves along the curve.

Available GUI Objects

The Lattice Mapping tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the lattice mapping tools.



These tools can be drag&dropped to a tool bar.

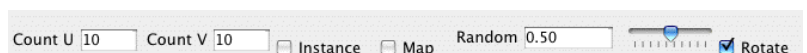
Distribute



Distribute over surface.

Select the target object and the lattice object. Define the Distribute tool options through the Control Bar and accept the tool via the View's pop-up menu or by clicking the Accept button on the Control Bar. This distributes copies or instances of the target object over the surface of the skeleton object.

The Distribute tool has the following options:



- Count U - Number of points is 'u' direction.
- Count V - Number of points is 'v' direction.
- Instance - If set, instances rather than copies.
- Map - If set, both maps and distributes, otherwise just distributes.
- Random - Randomize.
- Rotate - If set, rotates objects according to the surface normal.

Rotate



Make the first selected objects rotate about the last selected 'axis' object.

Select two objects: the object to be rotated and the rotation axis. Define rotation options through the Control Bar and accept the tool via the View's pop-up menu or by clicking the Accept button on the Control Bar. This creates a rotation animation.

The Rotate tool has the following option:

- Angle - Define rotation angle.



Map



Set up lattice mapping.

Select two or more objects. All but the last selected objects are mapped to the parameter space of the last selected object. Set mapping options through the control bar and accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Map tool has the following options:



- Mapping mode - Defines how the target object should be mapped.
 - Object
 - Pointwise
 - Selected Points
 - Look At point
- Weight - Defines weight for mapping. This is needed when mapping a point to multiple skeletons.

Multi Map



Map selected objects to lattice objects utilizing multi mapping.

Select two or more objects. The first selected objects are mapped to the parameter space of the last selected object. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

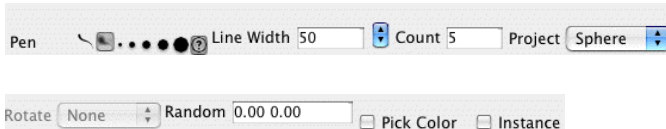
Paste



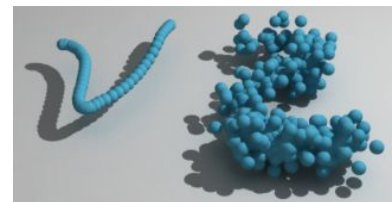
Copy and paste selected objects to a surface or into the 3D space.

Select the object(s) to copy. Drag over the desired positions with the left mouse button held down to create copies or instances.

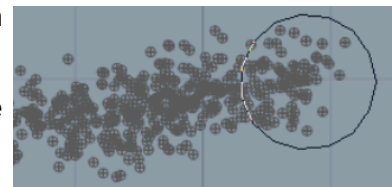
The Paste tool has the following options:



- Pen - paste single copies accurately or airbrush randomized object sets.
- Line Width - the radius of the airbrush pen measured as pixels. The brush is drawn as a circle when the mouse moves over a view window.
- Count - the number of object copies for the airbrush pen. The higher the value, the denser object cloud is pasted into the scene.



Pasting with a sharp pen and with an airbrush



Brush width is shown as a circle

- Project - defines how the mouse pointer position and created object copies are projected into 3D space:

- Sphere - objects are placed into a spherical volume around the mouse pointer coordinates on the input plane.
- Disc - objects are placed into a planar disc on the input plane.
- Surface - objects are projected into the underlying surfaces.



Comparison of spherical, flat disc and surface projected brushes

- Rotate - defines how surface-projected object copies are oriented:

- None - the original orientation is kept. Only the position changes.
- Normal - the z axis of the object space (the blue axis) is rotated to match the surface normal.
- Reflect - the z axis of the object space is rotated to match reflected directions.



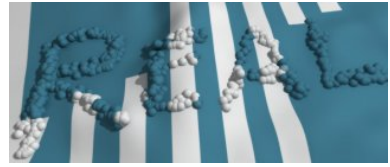
Hair brushed to three spheres from above using a fixed direction, surface normal's direction and reflected direction

- Random - randomizes the orientation of pasted copies. The first random value component controls how much the z axis direction of pasted copies is randomized. The second component randomizes the rotation around the z axis. For example, when pasting a tree object, the trunk should always point up, but the tree can rotate freely around its trunk. This can be achieved with a value pair "0 1".



A set of objects brushed spherically using a strongly randomized direction

- Pick Color - if checked, the color of the projection surface defines the color for created objects.
- Instance - paste instances instead of real object copies.



Underlying surface controls the color of pasted objects



Note

By holding down the Apple key while dragging, you can paste objects directly to the scene so that they start interacting with the tool. The tool updates the surface hit system every time you release the left mouse button. This means that by holding down the Apple and by Surface mode brushing objects with short LMB clicks, you can build cumulative piles of pasted matter. It is an interesting modeling approach and fun to play with.

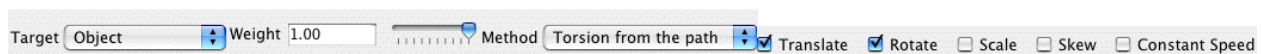
Path



Make the first selected objects follow last selected 'animation path' object.

Select two or more objects. The first selected objects will be moved along the last 'path' object. Define path options through the Control Bar and accept the tool via the View's pop-up menu or by clicking the Accept button on the Control Bar. This creates a path keyframe animation.

The Path tool has the following options:



- Target - Defines how the target object should be mapped.
 - Objects - The target is mapped as a rigid object
 - Points - Points of the target are mapped to the path. Path will deform the target.
 - Selected Points - same as above, but only for the selected points of the target object.
- Weight - Defines weight for mapping. This is needed when mapping a point to multiple skeletons.
- Translate - If set, the lattice object translates the target objects.
- Rotate - If set, the lattice object rotates the target objects.
- Scale - If set, the lattice object scales the target objects.
- Skew - If set, the lattice object skews the target objects.
- Method - controls how the path rotates the target object.
 - Torsion from the path - The target is rotated to follow the curve using the shortest rotation route between each two consecutive positions. This usually banks the target i.e. rotates it around the curve. Banking cumulates towards the end of the path.
 - No banking - A horizon axis is estimated from the path curve geometry, and all banking with respect to the horizon is eliminated.
 - Natural banking - The target is rotated to match the normal of the curve in non-cumulative manner.

- Cumulative banking - Same as Torsion from the path, but with a path level construct.
- Aeroplane Banking - A horizon axis is estimated from the curve. The banking behavior simulates aeroplane. The length of the horizon axis under the path level controls the amount of banking. A very long axis will eliminate banking.
- Constant Speed - If set, creates a constant speed path. Otherwise the target will accelerate and decelerate according to the point density of the curve.



Note

The path tool estimates the horizon plane from the curve geometry. If the curve is not planar, the estimate may be not what was intended. In that happens, just unmap the target, edit the horizon axis and map the target again to the path level, but this time with the default path options (torsion from the path, no constant speed) to avoid nested path levels.

Reconnect



Detach target object from existing lattices (skeletons) and attach it to newly selected objects.

Select the target object and new lattice objects for the target. Select Accept to detach the target objects from their original lattices and attach them to the selected lattice objects.

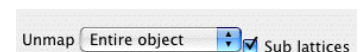
Unmap



Break mapping binding between the selected objects.

Select one or more target objects and the lattice object. Set Unmap tool options through the control bar. Select Accept to unmap objects from the lattice object.

The Unmap tool has the following options:

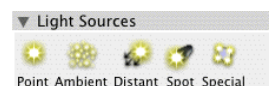


- Unmap - Define whether the tools should unmap only selected points or the entire object.
- Sub lattices - If set, also sub lattices are unmapped.

Light Source Creation Tools

Available GUI Objects

The Light Sources tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the light source creation tools.



These tools can be drag&dropped to a tool bar.

Ambient Light



Create an ambient light source.

Select Accept to create the light source to the camera position, or click the left mouse button in the View window to define the position for the light source. Move the mouse to define the light fall-off radius and click to accept.

See also: Models/Geometrics/Light Sources/Ambient Light.

Distant Light



Create a distant light source.

Click in a View window to define the first point for the light direction. Move the mouse to define the light direction and click to accept.

See also: Models/Geometrics/Light Sources/Distant Light.

Point Light



Create a point light source.

Select Accept to create the light source to the camera position, or click the left mouse button in the View window to define the position for the light source. Move the mouse to define the light fall-off radius and click to accept.

See also: Models/Geometrics/Light Sources/Point Light.

Special Light



Create a special light source. Any geometry inside the special light source object radiates light.

Select the objects to be converted into light sources. Select Accept from the View Window's popup menu or click the Accept Button.

See also: Models/Geometrics/Light Sources/Special Light.

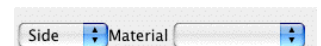
Spot Light



Create a spot light source.

Click the left mouse button in a View window to define the position for the light source. Move the mouse to define the light direction and click. Move the mouse and click to define the inner angle for the light cone. This is the angle in which the light sources casts rays of maximum brightness. Specify the outer angle for the light source. The brightness falls off with the angle and corresponds to zero with this angle.

The Spot Light tool has the following options:



- View - The Side view option defines the conical shape of the spotlight from the side. The Front view option places the spotlight exactly over the view camera. The user can define the inner and outer light cones by viewing along the spotlight axis.
- Material - Material defining volumetric properties

See also: Models/Geometrics/Light Sources/Spot Light.

Material Mapping Tools

The Material Mapping tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the material mapping tools.



These tools can be drag&dropped to a tool bar.

Average



Create 'Average map' by blending multiple materials with Scope channel.

Activate the tool. Select two or more mapping object to be averaged. Click Accept on the Control Bar or select Accept on the View window popup menu. The selected mapping objects are moved inside an average mapping level.

Cube



Create a cube mapping. Allows you to map six materials, one for each side.

Activate the tool. Specify options in the Control Bar. Define the mapping geometry (cube) by clicking two points on a View window. The materials for individual faces of the cube mapping can be set through the Property window.

The Cube Mapping tool has the following options:



- Define - How the mapping geometry is defined, either by two corners or center and corner.
- Map - Type of class to map. Either Material or Effect.
- Name of the material/effect - Define material or effect to be mapped.
- Environment Map - If set, creates an environment map.

See also: Models/Geometrics/Material Mapping Objects/Cube Mapping.

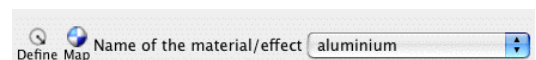
Cylinder



Map a material to an object using cylindrical mapping.

Activate the tool. Define the mapping geometry (cylinder) by clicking two or three points on a View window depending on the Define option.

The Cylinder Mapping tool has the following options:



- Define - How the mapping geometry is defined.
 - Center & Radius
 - Center & Axes
 - Corner & Corner

- Three points in arc
- Center & Corner
- Map - Type of class to map. Either Material or Effect.
- Name of the material/effect - Define material or effect to be mapped.

See also: Models/Geometrics/Material Mapping Objects/Cylinder Mapping.

Default



Map a material to an object using default mapping.

Activate the tool. Specify options in the Control Bar. Click Accept on the Control Bar or select Accept on the View window popup menu.

The Default Mapping tool has the following options:



- Material - Define material to be mapped.
- Mapping Type - Mapping method to use:
 - Surface - the material effect is mapped to the surface space of the actual geometry to be textured.
 - Identity - this method defines the material effects in identity space.
 - Null - this mapping method uses the current material space, as defined by the previous material mapping objects.
 - Camera - uses the current camera orientation to project material properties to geometric objects.
 - Custom - moves a selected channel (a vertex variable etc.) to the material mapping coordinate channel.

See also: Models/Geometrics/Material Mapping Objects/Default Mapping.

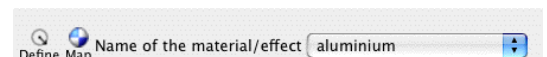
Disk



Map a material to an object using disk mapping.

Activate the tool. Specify options in the Control Bar. Define the mapping geometry (cylinder) by clicking two or three points on a View window depending on the Define option.

The Disk Mapping tool has the following options:



- Define - How the mapping geometry is defined.
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc

- Center & Corner
- Map - Type of class to map. Either Material or Effect.
- Name of the material/effect - Define material or effect to be mapped.

See also: Models/Geometrics/Material Mapping Objects/Disk Mapping.

Map2Obj



Convert mapped material properties to object properties.

Select one or more objects and one material mapping object. Activate the tool. Specify options in the Control Bar. Click Accept on the Control Bar or select Accept on the View window popup menu.



The Map2Obj tool has the following options:

- Target - Define properties of objects, points or selected points.
- Points - Define pointwise property for all geometry points.
- Selected Points - Define pointwise property for the selected geometry points.
- Objects - Define property for the object.
- Source - Source channel (material property) to be evaluated.
- Destination - Destination channel to be defined.
- Destination Material - Destination face material.

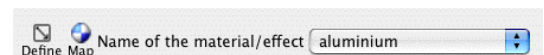
Parallel



Map a material to an object using parallel mapping.

Activate the tool. Specify options in the Control Bar. Define the mapping geometry (cube) by clicking two points on a View window.

The Parallel Mapping tool has the following options:



- Define - How the mapping geometry is defined, either by two corners or center and corner.
- Map - Type of class to map. Either Material or Effect.
- Name of the material/effect - Define material or effect to be mapped.

See also: Models/Geometrics/Material Mapping Objects/Parallel Mapping.

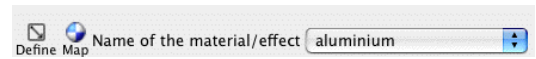
Pyramid



Map a material to an object using pyramid mapping.

Activate the tool. Specify options in the Control Bar. Define the mapping geometry (pyramid) by clicking two points on a View window.

The Pyramid Mapping tool has the following options:



- Define - How the mapping geometry is defined, either by two corners or center and corner.
- Map - Type of class to map. Either Material or Effect.
- Name of the material/effect - Define material or effect to be mapped.

See also: Models/Geometrics/Material Mapping Objects/Pyramid Mapping.

Sphere



Map a material to an object using spherical mapping.

Activate the tool. Specify options in the Control Bar. Define the mapping geometry (sphere) by clicking two points on a View window.

The Sphere Mapping tool has the following options:

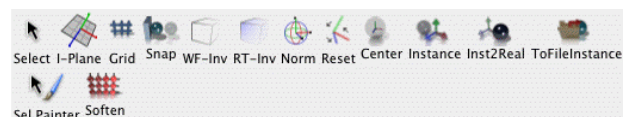


- Input - How the mapping geometry is defined.
 - Center & Radius
 - Corner & Corner
 - Center & Corner
- Map - Type of class to map. Either Material or Effect.
- Name of the material/effect - Define material or effect to be mapped.
- Environment Map - If set, creates an environment map.

See also: Models/Geometrics/Material Mapping Objects/Sphere Mapping.

Miscellaneous Tools

The Miscellaneous tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains some miscellaneous tools.



These tools can be drag&dropped to a tool bar.

Accept



Accept the active tool.

Cancel

 Cancel the active tool.

Center

 Moves the pivot point of selected object to the center of the bounding box.

Edit

 Toggle edit mode of the selected objects.

In edit mode the selected objects reveal their object specific handles allowing the user to activate object specific tools by dragging them. If edit mode is disabled, general transformation handles common to all geometric objects are shown.

Delete

 Delete the selected points.



Note

Not all the geometric objects support this tool. Many geometric objects define a fixed set of vertices.

Grid

 Modify the current grid or create new grids.

Select a grid and then enter two points on a View window to define the grid size and position. If you only want to change the grid position, click one point on a View window and then select Accept.

The Grid tool has the following option:

- Grid - List of available grids.

I-Plane

 Define the input plane.

Define a rectangle representing the input plane by clicking two points in a View window.

The I-Plane tool has the following option:

- Define - Method to be used for defining geometric data for the rectangle.
 - Corner-Corner - Define by two opposite corners

- Center-Corner - Define by center and corner

Inst2Real



Converts all instances and file instances to their real counterparts. The tool works recursively; you can select the root level of the hierarchy and apply the tool to convert all instances in the scene. It is naturally possible to convert a sub level or a single instance as well.



Note

The tool loads in all materials of the file instances, which it converts, and indexes the loaded materials to achieve unique names. The tool does this to avoid material name conflicts. For example, a converted file instance of a window object can include a material called 'glass' with regular glass properties, and another converted file instance of a wine glass may include another kind of glass material, with the same name but with a higher refraction index (crystal glass). Both object use the correct glass after conversion.

Invert



Invert point selection: make selected points unselected and unselected points selected.

Normalize



Normalize Object Space. The tool transforms the object space matrix (or desired components of it) to identity state. The geometric points are transformed with the inversion matrix so that their positions in the world space remains unchanged.

This tool is useful when one needs to animate transformed objects. Scaling and skewing is often introduced as a result of a normal modeling work flow. However, in animations, objects are easier to manage if their object spaces are world oriented.

The tool can be applied selectively to any of the following single transformation components:

- Translate
- Scale
- Rotate
- Skew

Orthogonal



Makes x,y and z dimensions of selected 3D particles perpendicular to each other.



Particles before and after Orthogonalization

Reset



Reset object handles to default.

Roll



Reorders XYZ directions of 3D particles. X becomes Y, Y becomes Z and Z becomes X axis. This has a visual effect when rendering 3D particle quads, because the quad surface is defined by the XY plane of each particle.

RT-Invisible



Toggle Ray Tracing invisibility of the selected objects.

Select



Select objects and their handles through the View windows.

You can make a selection by clicking the object or handle, by the drag box, or by the lasso selector. To use the drag box press the left mouse button down outside the object or handle and move the mouse so that the object or handle falls inside the drag box and then release the left mouse button. To use the lasso selector press down and hold the '4' key above the letter keys, press down the left mouse button and draw a freeform curve around the object or handle to select. The drag box and the lasso selector can be used to select multiple objects or handles. The object and handle selection modifiers are:

- Shift - Add to selection (OR)
- Ctrl - Intersection with previous selection (AND)
- Shift Ctrl - Invert selection (XOR)
- Alt - Subtract from selection
- 4 - Lasso selector in dragging

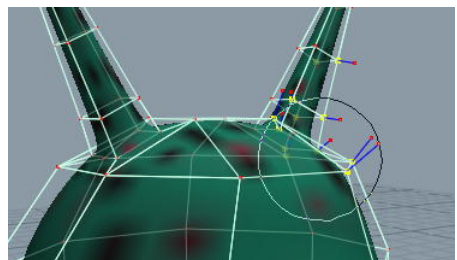
Selection Painter



Selection Painter is a tool which selects object editing handles (points, edges and faces) in a painting like manner. Selection happens simply by LMB-dragging a circular brush over the targets.

The tool can be used also for selecting entire objects. Object/handle painting is detected automatically from the already selected objects. If all selected objects are in editing mode, handle selection is painted. Otherwise (also when there is no selected objects yet), object selection is painted. Object selection painting is a useful feature when dealing with a large number of objects, such as curve modeled hairs.

The tool obeys the standard selection modifiers: Alt removes from the current selection and Shift adds points to the current selection.



Selecting point handles by painting

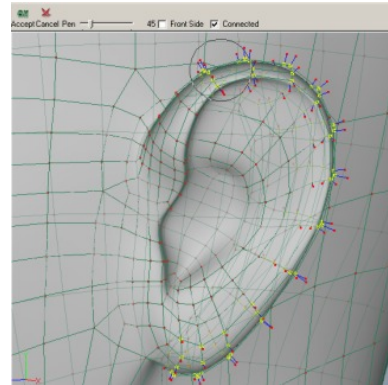
The size of the circular selection brush can be adjusted from the toolbar or with the mouse wheel.

The default shortcut for selection painter is Alt-Spacebar.

The Selection painter tool has an option for front side selections. The feature uses true geometric visibility to ensure that flipped SDS normals or open NURBS surfaces do not cause any problems.

The tool also supports connected selections. The connectivity feature limits the selection only to the neighborhood of already selected handles. Selection does not jump to disconnected surfaces that happen to be behind the desired target area. For example: select a tip of a finger and spread selection down the finger - other fingers will not interfere.

The Gradient option allows painting of soft selections. When enabled, the associated curve defines the distribution of the selection weight along the radius of the circular brush.



Connected selection selects the back side of the ear but not the skull behind the ear

Simulation



When a level object or an object derived from the level class is selected, the Simulation tool button appears on the Control Bar. You can activate and turn off simulation animation for the level.

Snap



The Snap tool snaps the selected objects to other objects. Snapping is based on ray/surface hit testing. Hit invisible objects, material maps and objects without 2D surface are not considered in snapping.

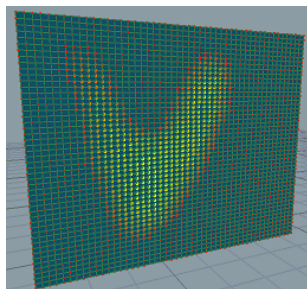
The snap tool options are the same as in the property gadget of the packer object (see Models/Geometrics section).

Soften



This tool softens handle selection values at the edges of the selection.

The tool is based on topological connections and therefore works only on geometric surfaces and curves where connectivity can be examined (NURBS, SDS, triset). By applying the tool multiple times, a wider softening can be achieved.



A hard edged lasso selection softened by applying the Soften tool multiple times

To File Instance



This tool converts the selected object(s) to a file instance object. Use the tool as follows:

- Select the desired object and start the tool.
- In the tool control bar, define the file name for saving the objects and geometry display method (bounding box/ accurate wireframe).
- Click Accept.



Note

The tool also collects all referred materials and saves the respective material chunk.

Volume Selector

Select points or objects by volume.

WF-Invisible



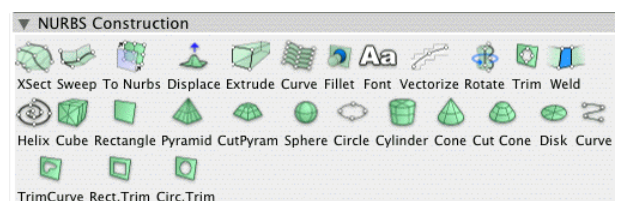
Make the object invisible in real time drawing (shaded OpenGL or wireframe).

NURBS Curves and Surfaces

NURBS Construction Tools

Available GUI Objects

The NURBS Construction tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the NURBS construction tools.



These tools can be drag&dropped to a tool bar.

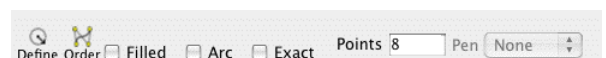
Circle



Create Nurbs circle.

Specify options in the Control Bar. Define the circle by clicking two or three points on a View window depending on the Define option.

The Circle tool has the following options:



- Define - How the circle is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Order - Order of the NURBS curve.
- Filled - If checked creates a disk, otherwise created a closed circular curve.

- Arc - If checked creates an arc.
- Exact - If checked creates an exact circle.
- Points - number of points along the circle.
- Density - number of extra curves.

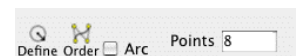
Circular Trim Curve



Create circular trim curve.

Specify options in the Control Bar. Define the circular trim curve by clicking two or three points on a View window depending on the Define option.

The Circular Trim Curve tool has the following options:



- Define - How the circle is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Order - Order of the trim curve.
- Arc - If checked creates an arc.
- Points - number of points along the circle.

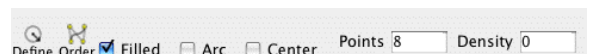
Cone



Create Nurbs cone.

Specify options in the Control Bar. Define the bottom circle of the cone by clicking two or three points on a View window depending on the Define option. Define the apex point by clicking on a View window.

The Cone tool has the following options:



- Define - How the cone is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner

- Order - Order of the NURBS surface.
- Filled - If not checked the top and bottom lids of the cone are not created.
- Arc - If checked creates a cone sector.
- Center - If checked, the apex will be centered over the bottom.
- Points - number of points along the bottom circle.
- Density - number of extra curves.

Cube



Creates a Nurbs cube. Specify options in the Control Bar. Define the cube by clicking two points on a View window.

The Cube tool has the following options:



- Define - How the cube is defined, either by two corners or center and corner.
- Order - Order of the NURBS surface.
- Filled - If not checked the top and bottom face of the cube are not created.
- U - number of points in the width direction.
- V - number of points in the height direction.

Curve



Create Nurbs curve.

Specify options in the Control Bar. If the Freehand option is set, then press down the left mouse button on a View window and draw the curve while holding the left mouse button down. Otherwise click a number of points on a View window and accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Curve tool has the following options:



- Filled - If checked creates a surface whose boundary is the defined freeform curve.
- Order - Order of the Nurbs curve or surface.
- Interpolate - If set the curve passes through the defined points.
- Freehand - Define curve by moving the mouse while holding down the left mouse button.
- Closed - Define closed curve or surface.
- Pen - When a Nurbs curve is selected define how a surface is created.
 - None - Creates a curve.
 - Sweep - The selected pen curve will be swept along the curve drawn. The pen object is rotated according to the curvature of the curve drawn.

- Rail - The selected pen curve will be swept along the curve drawn. The pen object is not rotated while sweeping.
- Rotate - Uses the pen object as an axis and rotates the new curve around it.

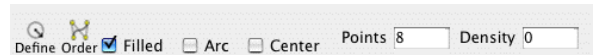
Cut Cone



Create Nurbs cut cone.

Specify options in the Control Bar. Define the bottom circle of the cut cone by clicking two or three points on a View window depending on the Define option. Define the top circle by clicking two points on a View window.

The Cut Cone tool has the following options:



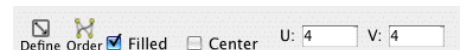
- Define - How the cut cone is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Order - Order of the NURBS surface.
- Filled - If not checked the top and bottom lids of the cut cone are not created.
- Arc - If checked creates a cut cone sector.
- Center - If checked, the top will be centered over the bottom.
- Points - number of points along the bottom circle.
- Density - number of extra curves.

Cut Pyramid



Create Nurbs cut pyramid. Define the bottom rectangle of the cut pyramid by clicking two points on a View window. Define the top rectangle point by clicking two more points on a View window.

The Cut Pyramid tool has the following options:



- Define - How the cut pyramid is defined, either by two corners or center and corner.
- Order - Order of the NURBS surface.
- Filled - If not checked the top and bottom face of the cut pyramid are not created.
- Center - If checked, the top will be centered over the bottom.
- U - number of points in the width direction.
- V - number of points in the height direction.

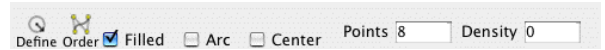
Cylinder



Create Nurbs cylinder.

Specify options in the Control Bar. Define the cylinder by clicking two or three points on a View window depending on the Define option.

The Cylinder tool has the following options:



- Define - How the cylinder is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Order - Order of the NURBS surface.
- Filled - If not checked the top and bottom lids of the cylinder are not created.
- Arc - If checked creates a cylinder sector.
- Center - If checked, the top will be centered over the bottom.
- Points - number of points along the bottom circle.
- Density - number of extra curves.

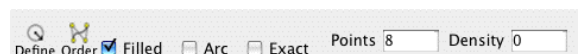
Disk



Create Nurbs disk.

Specify options in the Control Bar. Define the disk by clicking two or three points on a View window depending on the Define option.

The Disk tool has the following options:



- Define - How the Disk is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Order - Order of the NURBS curve.

- Arc - If checked creates an arc.
- Points - number of points along the disk perimeter.
- Density - number of extra curves.

Displace



Displace the selected Nurbs meshes with a bit map image.

Select the NURBS mesh you want to displace. Specify the displacement height and the image file in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Displace tool has the following options:



- Displacement - Defines the displacement height for the selected meshes.
- Image - Select bitmap file to be used for defining displacement.

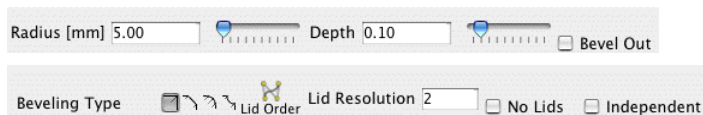
Extrude



Extrude selected curves with given bevelling. Creates an Extrusion Manager object.

Select the curve or curves to be extruded. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Extrude tool has the following options:



- Radius - Radius (size) of the beveling in millimeters.
- Depth - Extrusion depth.
- Bevel Out - Bevel in/out.
- Beveling Type - Defines how the top and bottom lids and the side mesh are welded with each other.
- Lid Order - NURBS order of the lid.
- Lid Resolution - Number of points in the top and bottom lids.
- No Lids - No top or bottom lids.
- Independent - Treat each selected object independently.

Fillet



Construct a fillet between surfaces, Nurbs curves, or trim curves.

Select two NURBS surfaces, or two or more NURBS curves. To build a fillet from the an open ended cylindrical mesh pointing to another mesh, select the target mesh first, then the cylindrical mesh. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Fillet tool has the following options:



- Trim - If set, created trim curves cut their parent surfaces.
- Radius - Fillet radius.
- Order - Order of the fillet.
- Resolution - Number of points in the 'U' direction of the fillet.

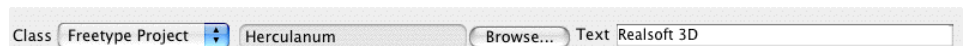
Font



Create Nurbs curves from outline fonts.

Select the font class. Set the font class specific options like the font name or font file. Type in the text to create font curves from. Click two points on a View window to define the baseline and size of the curves.

The Font tool has the following options:



- Class - Class of the font.
- Text - Enter text to be created.
- Render - Create rendering visible curves.

Note that each font class can have its own options.

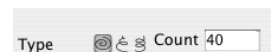
Helix



Create helix curves.

Set the helix type and the number of points on the Control Bar. Click the first point to define the center of the helix. Click the second point to define the starting direction of the helix curve. Move the mouse around the center point to define the shape of the helix curve and click to accept the tool.

The Helix tool has the following options:



- Type - Type of the curve to be created: Disk, Cone, or Cylinder.
- Count - Number of points in the curve.

Pyramid



Create Nurbs pyramid.

Specify options in the Control Bar. Define the bottom rectangle of the pyramid by clicking two points on a View window. Define the apex point by clicking on a View window.

The Pyramid tool has the following options:



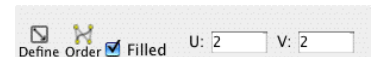
- Define - How the pyramid is defined, either by two corners or center and corner.
- Order - Order of the NURBS surface.
- Filled - If not checked the bottom face of the pyramid is not created.
- Center - If checked, the apex will be centered over the bottom.
- U - number of points in the width direction.
- V - number of points in the height direction.

Rectangle



Create Nurbs rectangle.

Specify options in the Control Bar. Define the rectangle by clicking two points on a View window. The tool has the following options:



- Define - How the rectangle is defined, either by two corners or center and corner.
- Order - Order of the NURBS surface.
- Filled - Whether to create a surface or a border line.
- U - number of points in the width direction.
- V - number of points in the height direction.

Rectangular Trim Curve



Create rectangular trim curve.

Specify options in the Control Bar. Define the rectangular trim curve by clicking two points on a View window.

The Rectangular Trim Curve tool has the following options:



- Define - How the trim rectangle is defined, either by two corners or center and corner.
- Order - Order of the trim curve.
- U - Number of vertices in the width (u) direction.

Rotate



Construct rotational nurbs surface by rotating the second curve about the first curve.

Select the NURBS curve to be used as an axis. Select the NURBS curve to be used as a profile curve. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Rotate tool has the following options:



- Order - Order of the rotation NURBS surface.
- Resolution - Number of points in the 'U' direction of the rotation surface.

Sphere



Create Nurbs sphere.

Specify options in the Control Bar. Define the sphere by clicking two points on a View window.

The Sphere tool has the following options:



- Input - How the sphere is defined, by center and radius, two corners, or center and corner.
- rder - Order of the NURBS surface.
- W - number of points in the width direction (along the equator).
- H - number of points in the height direction (from pole to pole).

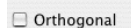
Sweep



The Sweep tool constructs a surface by moving a cross section curve along one or two rail curves.

Select the curve to be used as a cross section. Select one or two more curves to be used as rails. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Sweep tool has the following options:



- Orthogonal - Rotate the profile curve with the rail curve.

ToCurves



Create curves from surfaces of the selected objects.

Select the object to create curves from. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar. This creates a curve manager object whose attributes can be changed though the Property window.

The Curve tool has the following options:



- U Curves - Create curves in the 'U' direction of the parameter space.
- V Curves - Create curves in the 'V' direction of the parameter space.
- Curves - Number of curves to be created.
- Points - Number of points per curve.

ToNurbs



Convert selected objects to Nurbs surfaces of Bezier form.

Select the objects to be converted. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The To Nurbs tool has the following options:



- Density U - Number of Bezier patches per complexity unit in the 'U' direction.
- Density V - Number of Bezier patches per complexity unit in the 'V' direction.

Trim



Map the control points of a NURBS curve to the parameter space of a mesh.

Select one or more NURBS curves and one or more NURBS meshes. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar. The selected NURBS curves are projected onto the parent Nurbs mesh and corresponding trim curves are created on the NURBS meshes.

Trim Curve



Create Nurbs trim curve.

Select a NURBS mesh to trim. Specify options in the Control Bar. If the Freehand option is set, then press down the left mouse button on a View window and draw the trim curve over the NURBS mesh while holding the left mouse button down. Otherwise click a number of points on a View window and accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Trim Curve tool has the following options:



- Order - Order of the trim curve.
- Interpolate - If set the trim curve passes through the defined points.
- Freehand - Define the trim curve by moving the mouse while holding down the left mouse button.
- Closed - Define closed curve. Should be set.

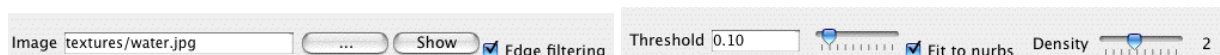
Vectorize



The Vectorize tool reads an image file, traces out contours from the image and generates a set of NURBS curves which represent the traced contours.

Select the bitmap image file to be vectorized. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The Vectorize tool has the following options:



- Image - Bitmap file to be vectorized.
- Edge Filtering - This should be checked unless the bitmap is a line drawing.
- Threshold - Edge threshold value. The smaller the value, the more edges are detected.
- Fit to nurbs - Cubic nurbs curves are fitted to edge polylines.
- Density - Point density of created curves.

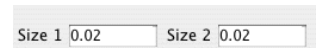
Weld



Weld single edges of two Nurbs meshes.

Select two NURBS meshes. Mark an edge from each mesh by dragging along the surface with the mouse. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar. A mesh is created to weld the selected edges together.

The Weld tool has the following options:



- Size 1 - Size of the weld at the first mesh.
- Size 2 - Size of the weld at the second mesh.

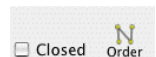
XSect



The Cross Section tool constructs a mesh through the selected curves.

Select two or more NURBS curves in the order you want them to appear in the NURBS mesh. Specify options in the Control Bar. Accept the tool by selecting Accept on a View window's popup menu or by clicking the Accept button on the Control Bar.

The XSect tool has the following options:



- Closed - Close the mesh in the 'V' direction.
- Order - Order of the NURBS surface.

NURBS Mesh Tools

Add



Insert new curve to the mesh.

Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to add and position both U and V directed curves. Release the mouse button to accept.

Add U directed curve Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to add and position a U directed curve. Release the mouse button to accept.

Add V directed curve Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to add and position a V directed curve. Release the mouse button to accept.

Break



Break the NURBS mesh from the given point.

Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to break the mesh by both U and V directed curves. Release the mouse button to accept.

Break by U directed curve Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to break the mesh by U directed curve. Release the mouse button to accept.

Break by V directed curve Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to break the mesh by V directed curve. Release the mouse button to accept.

CloseU



Toggle open/close the mesh in U direction.

Open in U direction Open the mesh in U direction.

Close in U direction Close the mesh in U direction.

CloseV



Toggle open/close the mesh in V direction.

Open in V direction Open the mesh in V direction.

Close in V direction Close the mesh in V direction.

Concatenate



Concatenate the selected two meshes.

Curves



Convert mesh to NURBS curves.

Creates curves that have the same control points as the mesh.

U and V curves Creates both U and V curves from the control points of the mesh.

V curves Creates V curves from the control points of the mesh.

DefineUV



Allocate and define texture UV coordinates for the mesh.

Uniform Allocate and define texture UV coordinates with uniform parametrization for the mesh.

Chord Length Allocate and define texture UV coordinates with chord length parametrization for the mesh.

Centripetal Allocate and define texture UV coordinates with centripetal parametrization for the mesh.

Delete



Delete selected points.

Edit



Choose how to select, either by points, curves of objects.

ElevU



Degree elevation in U direction.

Elevate with fixed shape The degree of the mesh is elevated by one. Elevation is done so that the shape does not change. For example, a linear mesh remains linear by its shape. However, you can later bend it smoothly because the higher degree allows it.

Elevate by smoothing the shape Degree elevation uses the control points to form as smooth shape as the higher degree allows.

ElevV



Degree elevation in V direction.

Elevate with fixed shape See U directed elevation above.

Elevate by smoothing the shape See U directed elevation above.

Fillet



Create fillet between two meshes.

Activates the Fillet tool.

Invert



Make the surface interpolate through the control points.

ReduceU



Reduce degree in U direction. Opposite to Degree elevation operation described above. Degree reduction usually changes the curve shape.

ReduceV



Reduce degree in V direction. Opposite operation of degree elevation.

Set Start



Set starting point for a closed mesh.

Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to the starting point of the mesh in both U and V directions. Release the mouse button to accept.

Set start in 'u' direction Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to the starting point of the mesh in U direction. Release the mouse button to accept.

Set start in 'v' direction Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to the starting point of the mesh in V direction. Release the mouse button to accept.

Sharpen



Insert sharp edge (crease) to the mesh.

Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to add and position both U and V directed creases. Release the mouse button to accept.

Add U directed crease Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to add and position a U directed crease. Release the mouse button to accept.

Add V directed crease Select a NURBS mesh object. Click and drag over the NURBS mesh object on a View window to add and position a V directed crease. Release the mouse button to accept.

Swap



Swap direction of the mesh.

Swap direction of the mesh in both U and V direction.

Swap U direction Swap direction of the mesh in the U direction.

Swap V direction Swap direction of the mesh in the U direction.

SwapUV



Swap direction of the mesh. The U direction becomes the V direction and vice versa.

Weight



Modify the weight of the selected control points.

Select the points to change the weight for. Activate the tool. Click on a view window. Move the mouse cursor up to increase weight or down to decrease weight. Click again to accept.

Weld



Weld two meshes to each other.

NURBS Curve Tools

Break



Break the curve at a specified point. A closed curve the curve is made open and the starting point is set to the specified point. An open curve is separated into two curves at the specified point.

Select a NURBS curve object. Click and drag over the NURBS curve object on a View window to set the breaking point. Release the mouse button to accept.

Close



Make closed curves open and open curves closed.

Open - Open the curve.

Open Exact - Open the curve so that its shape does not change.

Close - Close the curve.

Concatenate



Concatenate two or more Nurbs curves.

Delete



Delete selected points.

Distribute



Distributes curve points evenly (at an equal distance from each other) along the curve. The shape of the curve usually changes in this operation. If the curve is not smooth everywhere, changes can be remarkable.

If the curve is cubic, the number of points increases by 2 and parametrization changes to uniform.

Elevate



Degree elevation.

Elevate with Fixed Shape

The degree of the curve is elevated by one. Elevation is done so that the shape does not change. For example, a linear curve remains linear by its shape. However, you can later bend it smoothly because the higher degree allows it.

Elevate by Smoothing the Shape

Degree elevation uses the control points to form as smooth shape as the higher degree allows.

Extrude



Extrude selected curves with given bevelling.

Activate the Extrude tool with the selected curves.

Invert



Make the curve pass through its knot points.

InvKin



Move the selected point with Inverse Kinematics.

Select a point from a Nurbs curve. Activate the InvKin tool. Set options on the Control Bar. Click on a View window. Move mouse to move the selected point. Click again to accept.

Measure



Measures the length of the curve (in meter scale). The result is printed into the status window and to the info window.

New Point



Insert new point to the curve. The shape of the curve is not changed.

Select a NURBS curve object. Click and drag over the NURBS curve object on a View window to add and position the new point. Release the mouse button to accept.

Parallel



Creates a parallel curve from the selected curve. The created curve travels through a point, which user defines by a mouse click.

The offset distance is added in the view plane of the window where the user clicked the offset point. This means that it is usually best to apply the tool from the same viewing direction from which the curve was created.



Parallel curves created for some curve shapes

The created parallel curve has an equal number of points with the original curve. Second order parallel curves (polylines) follow the original curve exactly at the defined distance. For higher degree curves, such as cubic curves, this is generally not possible. The control polygons are parallel but the actual curves may get closer or further apart at strongly bent curve sections.

Param



Set curve parametrization.

Uniform - Set uniform parametrization for the curve.

Chordlength - Set chord length parametrization for the curve.

Centripetal - Set centripetal parametrization for the curve.

Reduce



Reduce degree in U direction. Opposite to Degree elevation operation described above. Degree reduction usually changes the curve shape.

Rotate



Construct rotational nurbs surface by rotating the second curve about the first curve.

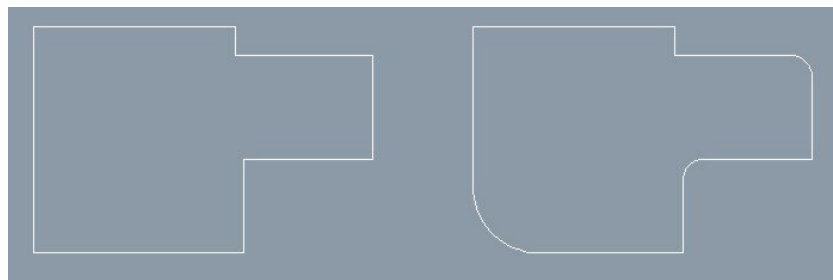
Activate the Rotate tool with the selected curves.

Round



Round the control polygon of selected curves at selected points.

Number of points used for rounding and rounding radius can be defined.



Original control polygon at left, rounding applied to some vertices at the right side

Sharpen



Add a sharp point on the selected curve.

Select a NURBS curve object. Click and drag over the NURBS curve object on a View window to add and position the new sharp point. Release the mouse button to accept.

Start



Set curve start point.

Select a closed NURBS curve object. Click and drag over the NURBS curve object on a View window to set the start point. Release the mouse button to accept.

Swap



Swap the direction of the curve.

Sweep



Construct a surface by moving a cross section curve along one or two rail curves.

Activate the Sweep tool with the selected curves.

ToBezier



Convert to Bezier curve.

Trim



Trim the parent mesh.

Select a Nurbs curve whose parent is a Nurbs mesh. Activate the Trim tool. The selected Nurbs curve is projected onto the Nurbs mesh and corresponding trim curve is created on the Nurbs mesh.

Weight



Modify the weight of the selected points.

Select the points to change the weight for. Activate the tool. Click on a view window. Move the mouse cursor up to increase weight or down to decrease weight. Click again to accept.

XSect



Construct a mesh through the selected curves.

Activate the XSect tool with the selected curves.

TrimNurbs Tools

Invert



Set to trim out inside, reset to trim out outside.

Toggle the Invert attribute of the trim curve, whether the surface points inside or outside the curve are trimmed out.

To3D



Map the selected trim curves to regular 3D nurbs curves.

Trim



Trim the surface.

Toggle the Cut attribute of the trim curve, whether to trim the parent mesh or not.

Weld



Create a weld surface through the two selected trim curves.

Activates the Weld tool with the selected trim curves.

Plant Tools

Branch



Adds a new sub branch object to each selected branch or plant root object. You can use hierarchal branch structures to define growth behaviors, which change a lot between generations: trunk, main branches, small branches etc.

Leaf



Adds a new leaf object into each selected branch or plant root object.

Plant



Creates a fractal plant's seed object with a sub branch object. To use the tool:

- Activate the Plant Root tool from the tool bar or from the pull down menu.
- Click on a view window to define the point where the plant grows.
- Move the mouse upwards to define the direction and speed of the growth. Click second time to finish the tool.

Randomize



Sets a new random seed value to selected plant root objects. If plants have randomized properties, each plant becomes unique.



These small trees have the same growth properties but different random seeds

Root



Adds roots to the selected plant. To use the tool, select a plant (or several plants) and click the button. The tool does not need any mouse clicks.

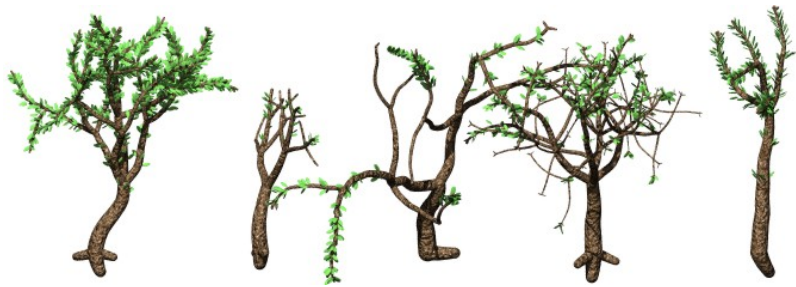
The tool adds a root sub object to each selected plant. A root object is similar to the branch object, except that it grows downwards. The geometry type of the root is initialized to match the first branch object.

Variate



Adds a random change to all growth properties of the selected plant components and their sub objects.

Applying the tool a couple of times usually changes the look of the plants quite much.



A plant and some variations created from it

Reset



Resets attributes of the selected plant component(s) back to the default state. This tool does not work hierarchically. Applying it to a plant seed level resets only the seed attributes - sub branches and leaves will not change.

Subdivision Surface Tools

Subdivision Creation Tools

The SDS tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains many SDS creation tools.



These tools can be drag&dropped to a tool bar.

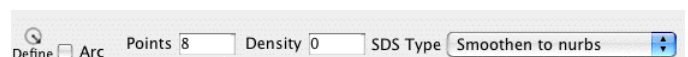
Circle



Create SDS circle.

Specify options in the Control Bar. Define the circle by clicking two or three points on a View window depending on the Define option.

The SDS Circle tool has the following options:



- Define - How the circle is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Arc - If checked creates an arc

- Points - number of vertices along the circle
- Density - number of extra curves
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

Cone



Create SDS cone.

Specify options in the Control Bar. Define the bottom circle of the cone by clicking two or three points on a View window depending on the Define option. Define the apex point by clicking on a View window.

The SDS Cone tool has the following options:



- Define - How the cone is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Arc - If checked creates a cone sector
- Points - number of vertices along the bottom circle
- Density - number of extra curves
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

Cube



Create SDS cube.

Specify options in the Control Bar. Define the cube by clicking two points on a View window.

The SDS Cube tool has the following options:



- Define - How the cube is defined, either by two corners or center and corner.
- Filled - If not checked the top and bottom face of the cube are not created.
- U - number of vertices in the width direction
- V - number of vertices in the height direction
- W - number of vertices in the depth direction
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

Cut Cone



Create SDS cut cone.

Specify options in the Control Bar. Define the bottom circle of the cut cone by clicking two or three points on a View window depending on the Define option. Define the top circle by clicking two points on a View window.

The SDS Cut Cone tool has the following options:



- Define - How the cut cone is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Arc - If checked creates a cut cone sector
- Points - number of vertices along the bottom circle
- Density - number of extra curves
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

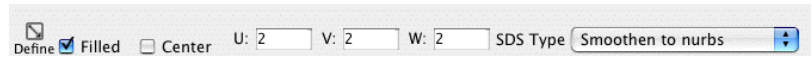
Cut Pyramid



Create SDS cut pyramid.

Define the bottom rectangle of the cut pyramid by clicking two points on a View window. Define the top rectangle point by clicking two more points on a View window.

The SDS Cut Pyramid tool has the following options:



- Define - How the cut pyramid is defined, either by two corners or center and corner.
- Filled - If not checked the top and bottom face of the cut pyramid are not created.
- Center - If checked, the apex will be centered over the bottom.
- U - number of vertices in the width direction
- V - number of vertices in the height direction
- W - number of vertices in the depth direction
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

Cylinder



Create SDS cylinder.

Specify options in the Control Bar. Define the cylinder by clicking two or three points on a View window depending on the Define option.

The SDS Cylinder tool has the following options:



- Define - How the cylinder is defined. The choices are:
 - Center & Radius
 - Center & Axes
 - Corner & Corner
 - Three points in arc
 - Center & Corner
- Arc - If checked creates a cylinder sector
- Points - number of vertices along the bottom circle

- Density - number of extra curves
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

Pyramid



Create SDS pyramid.

Specify options in the Control Bar. Define the bottom rectangle of the pyramid by clicking two points on a View window. Define the apex point by clicking on a View window.

The SDS Pyramid tool has the following options:



- Define - How the pyramid is defined, either by two corners or center and corner.
- Filled - If not checked the bottom face of the pyramid is not created.
- Center - If checked, the apex will be centered over the bottom.
- U - number of vertices in the width direction
- V - number of vertices in the height direction
- W - number of vertices in the depth direction
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

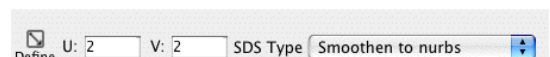
Rectangle



Create SDS rectangle.

Specify options in the Control Bar. Define the rectangle by clicking two points on a View window.

The SDS Rectangle tool has the following options:



- Define - How the rectangle is defined, either by two corners or center and corner.
- U - number of vertices in the width direction
- V - number of vertices in the height direction

- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

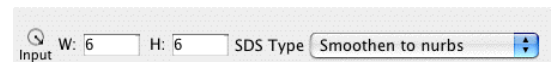
Sphere



Create SDS sphere.

Specify options in the Control Bar. Define the sphere by clicking two points on a View window.

The SDS Sphere tool has the following options:



- Input - How the sphere is defined, by center and radius, two corners, or center and corner.
- W - number of vertices in the width direction (along the equator)
- H - number of vertices in the height direction (from pole to pole)
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

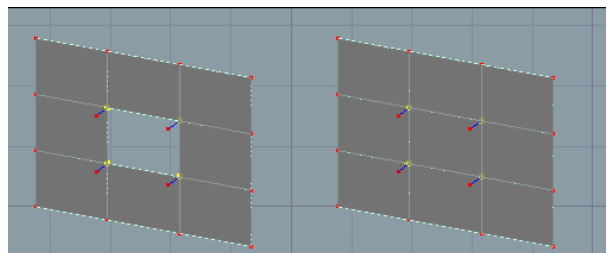
Subdivision Editing Tools

Add



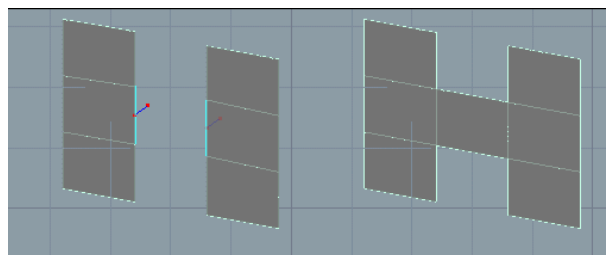
Add Face

Creates a new face that connects the selected vertices. The order in which the vertices are selected determine their order in the face.



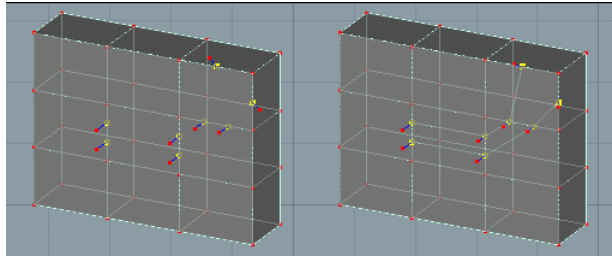
Add Face by Connecting Edges

Creates a face between two selected edges.



Add Edge

Creates new edges between the selected vertices of a face thus dividing the face. The number of selected vertices in a face must be even and the longest contiguous selection must contain exactly half of the selected vertices in that face.



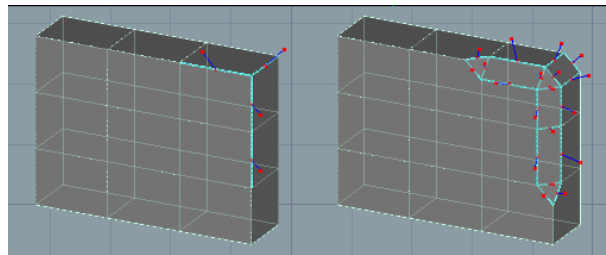
Bevel



Bevel edges or vertices

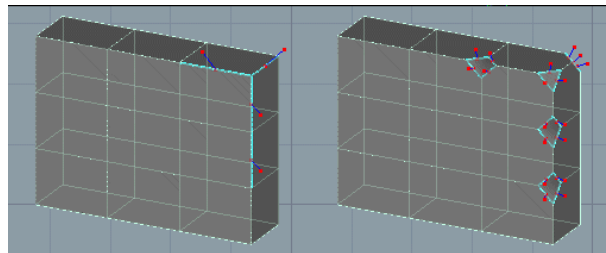
Bevel Edge

The first click on a view window starts the beveling. The distance of the cursor from the first click position defines the amount of beveling. The second click finishes the tool.



Bevel Vertex

The first click on a view window starts the beveling. The distance of the cursor from the first click position defines the amount of beveling. The second click finishes the tool.



Crease

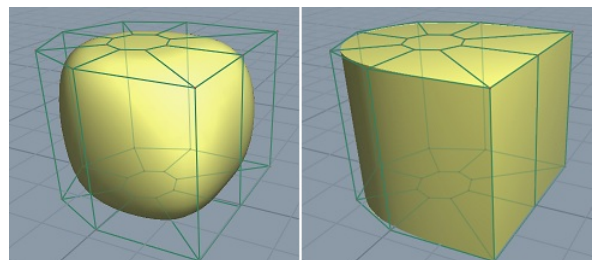
This tool examines SDS geometry and sets sharp creases to all points/edges, where the angle between connected faces exceeds the given limit.

In point/edge/face mode, the tool changes only selected areas. In object mode, the whole object is considered.



Note

The tool only adds creases. It does not remove user defined creases. Semi sharp or free creases are not changed to sharp.



Set edge and point creases automatically by a threshold angle

The tool has the following controls:

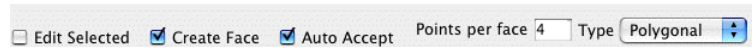
- **Angle** - the threshold angle. In edge crease creation, surrounding faces must exceed the limit angle at both ends of the connecting edge. A sharp point crease is set when all faces connect to the point's average normal with an angle that is greater than the limit. For example, a cone like surface usually gets a point crease in the apex, although surrounding cone side faces join to each other smoothly. That happens because each face has a sharp angle with the apex direction.
- **Target** - selects if point, edge or both crease types should be considered. Often the threshold angle should be different for edge and point creases.

Create Subdivision Object



The interactive SDS (SubDivision Surface) creation tool can be used to create SDS vertices and faces or to add them to an existing SDS object. The SDS creation tool is in the 'SDS' section of the default toolbar.

The creation tool has the following options:



- **Edit Selected** - If checked adds new vertices and faces to the selected SDS object, otherwise creates a new SDS object
- **Create Face** - If checked connects the new vertices into faces, otherwise creates separate vertices
- **Auto Accept** - If checked automatically accepts a face when Points per face vertices have been entered.
- **Points per face** - How many vertices go into a face when the Auto Accept option is selected
- **Type** - SDS Object type, e.g. polygonal

New vertices can be added by clicking on a view window. Holding down the 'Alt' modifier key while clicking adds the nearest existing vertex to the new face.

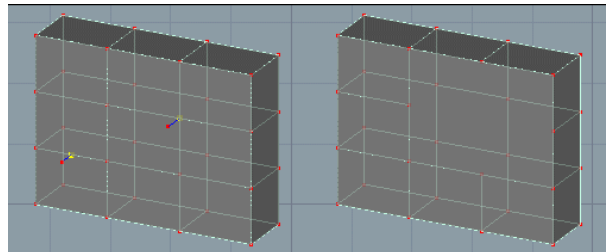
- ✓ The **Accept Face** button can be used to accept a face when the Auto Accept option is not used or when accepting a face with fewer vertices than in the Points per face control.

Delete

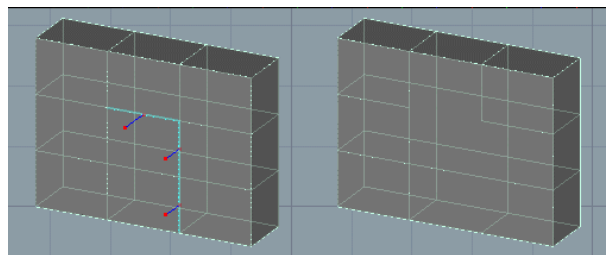


Selected:

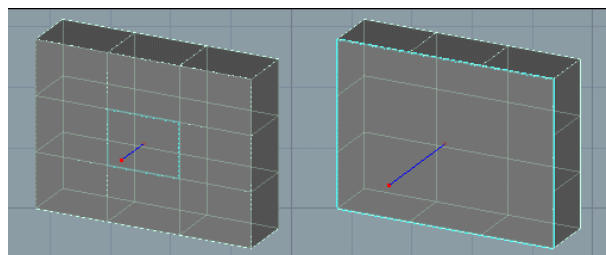
In the vertex selection mode removes selected vertices. If a selected vertex is a shared corner of multiple faces then these faces are merged.



In the edge selection mode merges the faces that share a selected edge.



In the face selection mode the selected faces are merged with their neighbors.



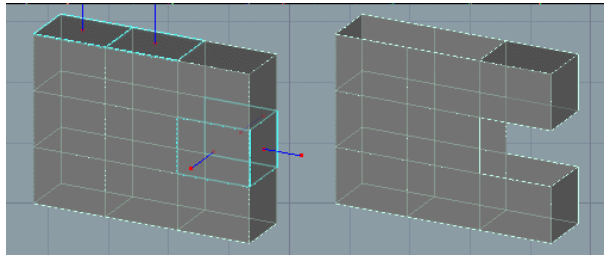
Orphan Vertices: Deletes all vertices that are not part of any face.

Destroy Faces



Destroy Faces - Destroys selected faces and vertices that were connected only to the destroyed faces.

Destroy Faces, keep orphan vertices - Destroys selected faces but does not delete any vertices.



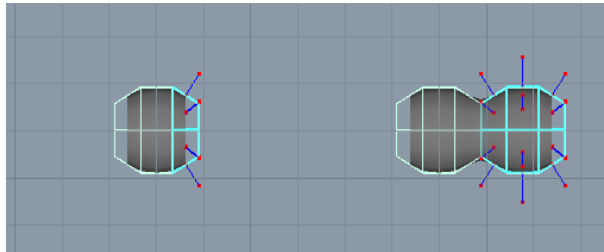
Duplicate and Mirror



Duplicates those parts of the SDS geometry that do not form a closed surface. The duplicated geometry is mirrored about the open edges loops (holes or outlines).

If there are more than one open edge loop, then selecting an edge or face at the open edge loop selects the edge loop about which to mirror.

The mirrored geometry is connected to the edge loop. The vertices at the edge loops are not duplicated.



Duplicate and Mirror with a selected edge loop

Duplicate Faces



Copies all the faces. If the object has open edges then the open edges of the original faces and those of the copied faces are connected with new faces. The copied faces become selected so that they can be readily transformed, e.g. moved apart.

Duplicate Selected Edges

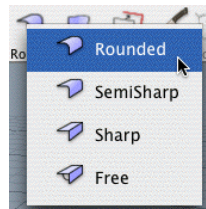


Duplicates selected edges and creates new faces to connect the duplicates with the original edges. The duplicates are selected so that they can be readily modified, e.g. moved to a new position.

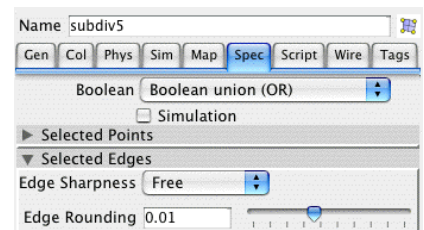
Edge Rounding

Sets rounding of the selected edges. Possible values are:

- Rounded
- SemiSharp
- Sharp
- Free



Set rounding of the selected edges



When edge rounding is set to Free, the rounding radius can be adjusted through the Spec tab of the property window

Extrude



Extrude selected faces or vertices

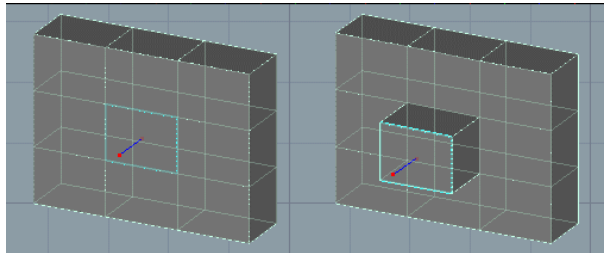
Extrude Face

This tool has the following options:



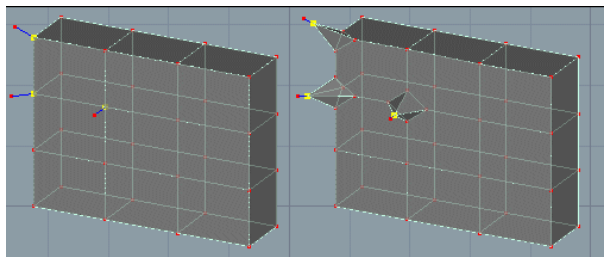
- Region - Groups: adjacent faces are extruded as a single region, Separately: each face is extruded separately
- Normals - Group: all vertices in a face group are moved in the direction defined by all the faces in the group, Vertex: each vertex is moved in the direction of its normal, Selected face vertex: the direction of each vertex is the average of the normals of the selected faces that the vertex belongs to.
- Leave Floor - If checked faces are added where the selected faces were originally.
- Sections - The extrusion is divided into this many sections.

The first click on a view window starts the extrusion. Moving the mouse in the vertical direction adjusts the amount of extrusion. The second click finishes the tool.



Extrude Vertex

The first click on a view window starts the extrusion. Moving the mouse in the vertical direction adjusts the amount of extrusion and horizontal movement adjusts the size of the base of the extrusion. The second click finishes the tool.



Faces



Activates the interactive Subdivision Creation Tool described in the beginning of this chapter.

Fill Holes

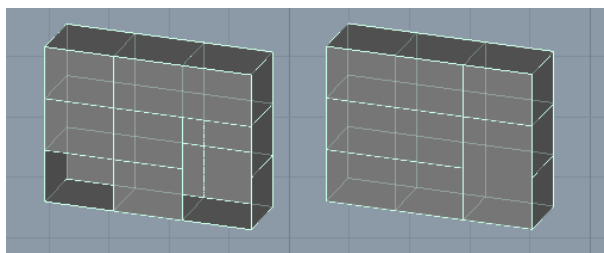


Adds new faces where there are loops of edges that are connected to one face only. These edges define the holes that can be filled.

This tool has the following options:



- Max Edges - the maximum number of edges around a hole to be filled. Only holes that have don't have more than this many bordering edges are filled.
- Selected - If checked, only holes in selected regions are filled.

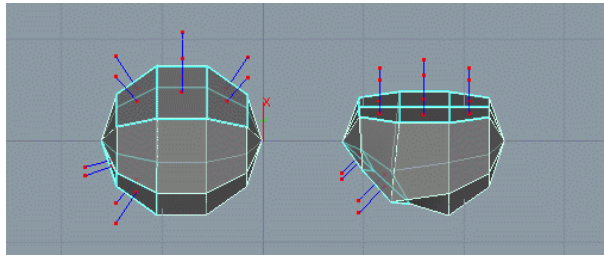


Flatten



In face mode flattens selected face regions separately.

In vertex or edge mode flattens all selected vertices or edges as one group regardless of topology.



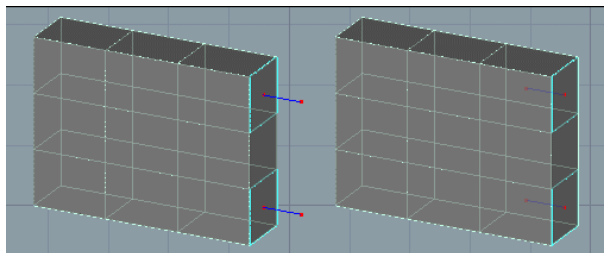
Flatten with two selected face regions

Flip Faces



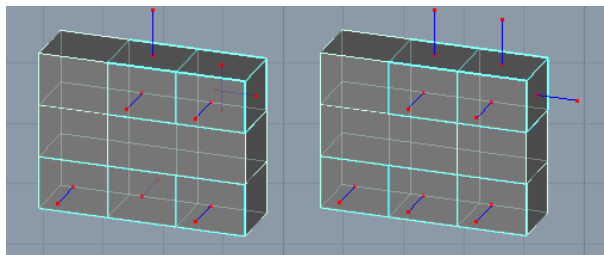
Separately

Flips the face normals of the selected faces to the opposite direction.



Regions

Flips the opposing face normals in each selection region so that all faces normals in that region will point to the same side. The majority of faces in the region defines to which side the normals will point. If none of the faces is selected then the whole object is considered as selection regions defined by one or more groups of adjacent faces.



Freehand Knife



Lets the user draw a freeform curve or a line segment that is used to cut edges of the underlying subdivision object. The edges are cut by adding new vertices where the knife hits the edges.

The knife tool has the following options:



- **Selected** - If checked only selected edges can be cut, otherwise all edges can be cut.
- **Add Edges** - If checked connects the new vertices with new edges to divide the faces into smaller faces, otherwise creates only the vertices where the knife cuts the edges.
- **Select** - Select which faces will be selected after cutting.
 - **No selection** - No faces will be selected
 - **Inside** - The faces inside the freehand curve will be selected
 - **Outside** - The faces outside the freehand curve will be selected

The selection is based on the center point of the face (whether it is inside or outside the curve).

- Freehand - If checked a freehand curve can be drawn, otherwise a single line segment can be drawn.

Hide/Show



Hide selected faces or show all faces

Hide Selected Faces - Hides selected faces in wireframe display.

Show All Faces - Displays all faces.

Init UV

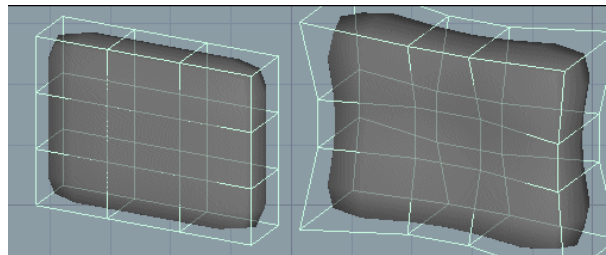


Calculates default values for the currently selected UV coordinate channel. The direction of each vertex from the center (average) of all vertices defines the UV value for that vertex.

Invert



Makes the surface go through the original control polygon. This is true only for non-polygonal SDS objects.



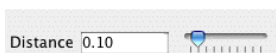
SDS cube inverted

Join Faces

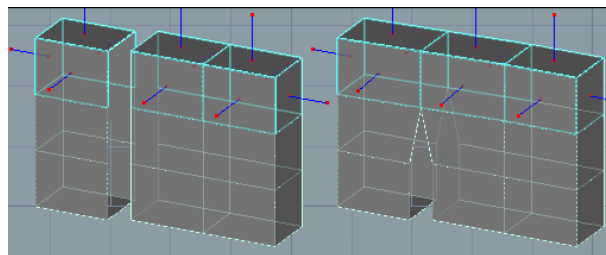


Merges near-enough pairs of open edges thus joining the faces. If there are selected faces or edges then only these edges will be considered. If no face or edge is selected then all edges will be considered.

This tool has the following option:



- Distance - the maximum distance between edges to be connected. Edge pairs that are further apart will not be considered.

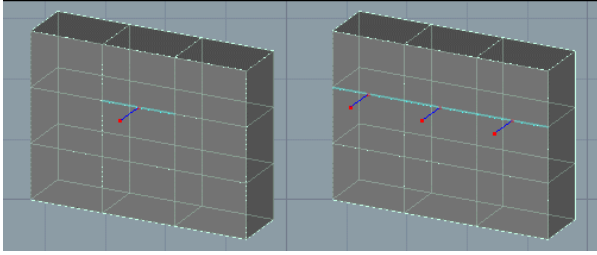


Loop

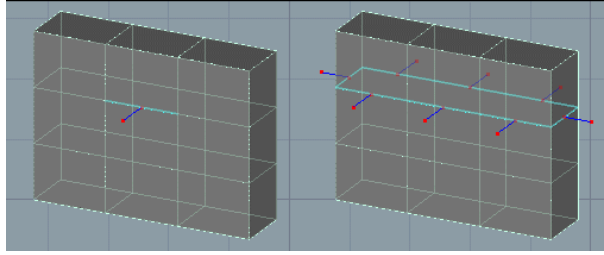


Expands the current edge selection either to contiguous edges or parallel edges. This tool uses the following modifiers keys:

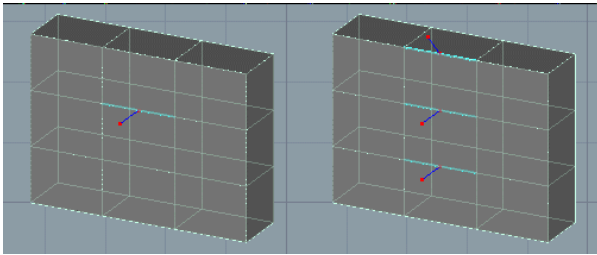
- Shift - Expands the selection as far as there are edges that meet the selection criteria.
- Apple - Selects edges that are parallel to the currently selected edges.



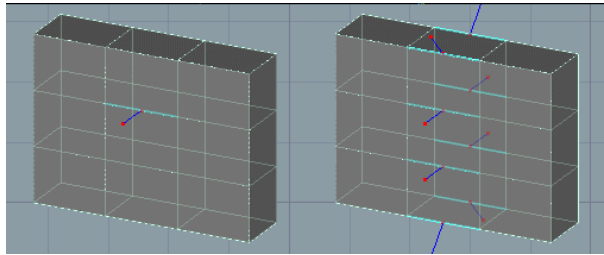
Edge selector tool used without modifier keys



Edge loop selector tool used with the <Shift> modifier key



Edge loop selector tool used with the <Apple> modifier key



Edge loop selector tool used with the <Apple> and <Shift> modifier keys

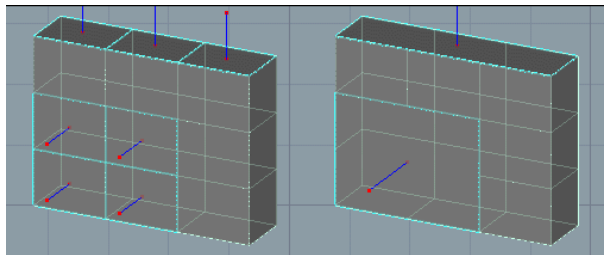
Merge



Merge Faces, Edges or Vertices

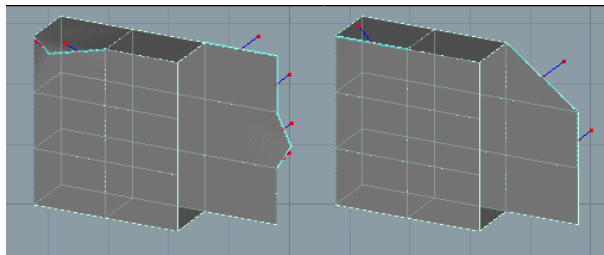
Merge Faces

Merges adjacent selected faces into a single face.



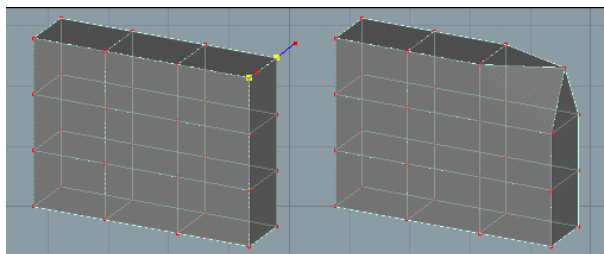
Merge Edges

Merges two or more contiguous edges into one. All these edges must be either connected only to a single face or go along a border between two faces.



Merge Vertices

Merges two or more selected vertices into one.



Merge Objects



Merge Objects - Merges all selected SDS objects into a single SDS object.

Merge Objects, Selected Faces - Merges the first selected SDS object and the selected faces from the other selected SDS objects into a single SDS object.

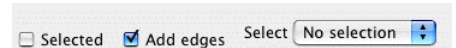
Object Knife



Lets the user cut SDS objects with the surface of a tool object. The edges are cut by adding new vertices where the tool object's surface hits the edges.

Select one or more SDS objects and one tool object (selected last). Set the tool options on the Control Bar and accept the tool.

The object knife tool has the following options:



- **Selected** - If checked only selected edges can be cut, otherwise all edges can be cut.
- **Add Edges** - If checked connects the new vertices with new edges to divide the faces into smaller faces, otherwise creates only the vertices where the knife object cuts the edges.
- **Select** - Select which faces will be selected after cutting.
 - **No selection** - No faces will be selected
 - **Inside** - The faces inside the tool object will be selected
 - **Outside** - The faces outside the tool object will be selected

The selection is based on the center point of the face (whether it is inside or outside the tool object).

Separate



In face mode disconnects the selected faces from the rest of the SDS object. The separated faces remain selected so that they can be e.g. moved to a new place.

In edge mode separates the model along selected edges.

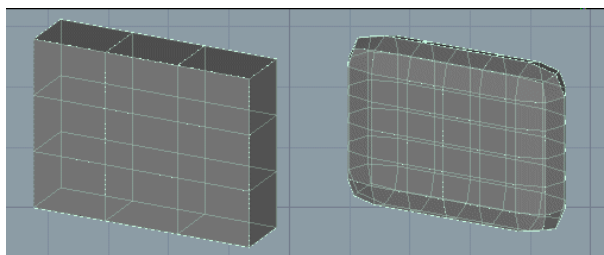
In vertex mode each connected face gets its own copy of the selected vertices.

Smooth



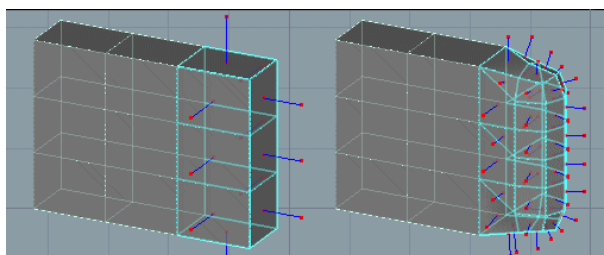
Smooth Whole Object

The subdivision algorithm is used to create a smoother SDS object with more vertices and faces.



Smooth Locally

The subdivision algorithm is used to create a smoother SDS object from the selected faces. This smoothed object is then joined with the unselected faces.



Smoothen/Flatten

Smoothen or flatten selected shading normals



This tool can be applied to polygonal SDS objects, which have the Phong shading option set. To use it, set the target object to the edit mode, select some faces/edges/points and apply the tool. Clicking the tool icon toggles between smooth and sharp shading normals. From the side menu, the desired alternative can be chosen directly as follows:

Smoothen selected shading normals

Sets the phong shading normal of every face passing through a selected point to the common average normal. Phong shading interpolates smoothly over the selected points.

Flatten selected vertex normals

Sets the phong shading normal of every face passing through a selected point to the geometric normal of the polygon corner in question (computed from the two surrounding polygon edges). Phong shading does not interpolate the normal smoothly but uses a private normal per face per selected point.

Split Edge

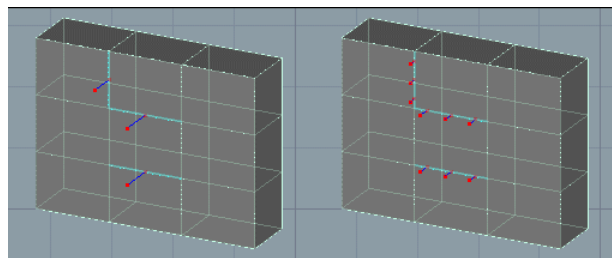


Adds one or more new vertices along each selected edge thus dividing the edges into two or more pieces. The new vertices are selected.

This tool has the following option:

Splits

- Splits - how many new vertices to add to each selected edge.



Split Edge tool with 2 splits used

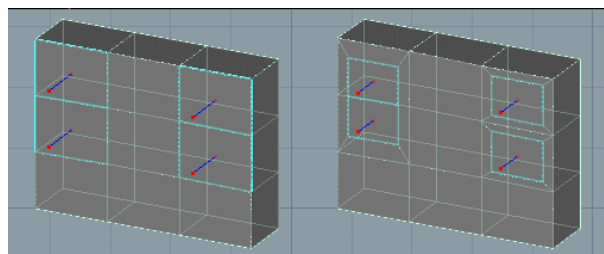
Subdivide



This tool has the following options:

Region Sections

- Region:
 - Groups: adjacent faces are subdivided as a single region
 - Separately: each face is subdivided separately
- Sections - The subdivision is divided into this many sections.



Subdivided faces with Region set to Groups and Separately

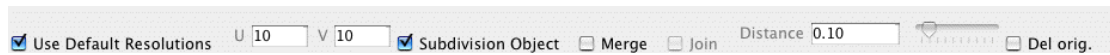
The first click on a view window starts the subdivision. Moving the mouse in the vertical direction adjusts the amount of subdivision. The second click finishes the tool.

To SDS



Converts selected objects to subdivision surface objects or triset objects.

The ToSDS tool has the following options:



- Use Default Resolutions - Check this to use default number of faces.
- U - Number of faces in the U direction. Can be set if Use Default Resolution is unchecked.
- V - Number of faces in the V direction. Can be set if Use Default Resolution is unchecked.
- Subdivision Object - Check this to make Subdivision object instead of triset.
- Merge - Check this to merge all converted objects into a single Subdivision object.
- Join - Check this to join faces that have edges near enough.
- Distance - Maximum edge distance for joining faces.
- Del orig. - Check this to delete the originally selected objects after conversion.

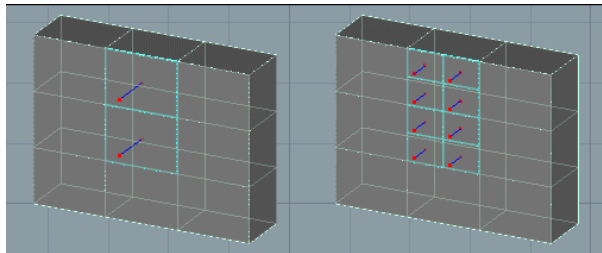
Tri/Quad



Divide faces to quads or triangles

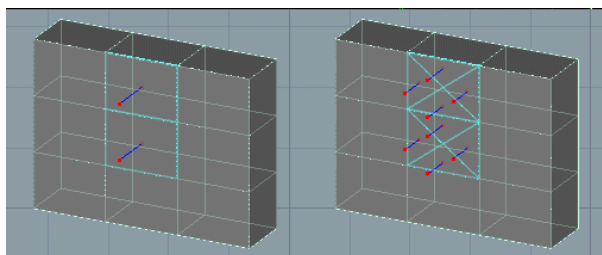
Divide Faces to Quads

Divides the selected faces to quads.



Divide Faces to Triangles

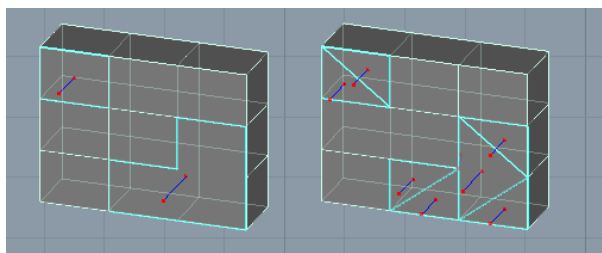
Divides the selected faces to triangles. Adds a new point to the middle of the face.



Triangulate Selected faces

Divides the selected faces to triangles without adding any new points.

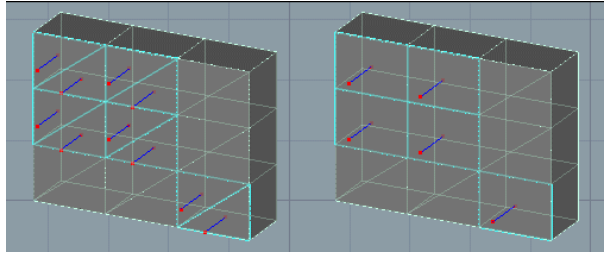
- Triangulate Quads - If checked, also quad faces will be divided into triangles.
- Convert to Quads - If checked, the triangles will be converted to quads where possible.



Triangles to Quads



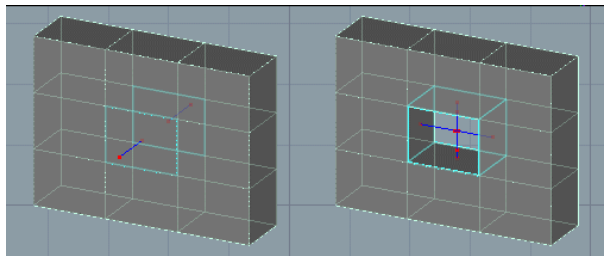
Combines selected triangular faces in pairs into quads.



Tunnel



Creates a tunnel between two selected areas. Each area can contain one or more adjacent faces.



Unwrap



Unwraps the geometry of the SDS object. Bumps and peaks become smoother, and points more evenly distributed. Usually unwrapping also shrinks the object, which side effect can be compensated by resizing the object suitably after the operation.

In edit mode, only the selected areas change.

Unwrap UV



Unwraps the current UV channel. UV vertices distribute more evenly.

UV Set



Selected faces

Creates a UV Set object containing the selected faces.

Cubic UV Sets

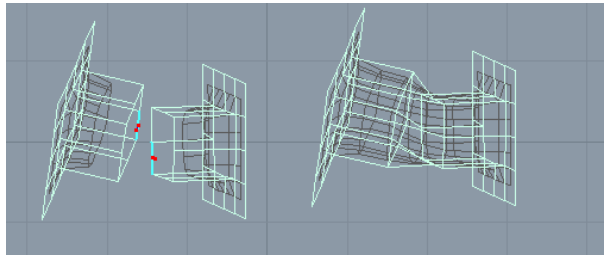
Creates a UV Set object for each main direction (Top, Bottom, Left, Right, Front, Back) containing the selected faces whose normals point into that direction. The directions are determined by the object's coordinate system.

Weld



Welds two open edge loops together by adding faces in the gap.

If there are more than two open edge loops, then selecting an edge or face at the open edge loop selects the edge loop to weld.



Weld with a two selected edge loops

SDS UV Set tools

Many of the SDS specific tools are also supported by the UV Set class. In addition to this, the UV Set specific control bar shows the following tools.

Connect



Connects selected areas by parent topology. Connects UV Set faces whose parent faces are already connected.

UV From View



Defines UV of the selected faces or vertices by their relative position on the current View window.

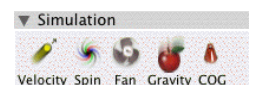
To Face Material



Copies UV Sets to the parent SDS object as face materials. UV Set object's child mapping objects define the face material names. The UV values for the face materials are taken from the UV Sets.

Simulation Tools

The Simulation tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the simulation tools.



These tools can be drag&dropped to a tool bar.

COG



Define center of gravity for the selected objects.

On a View window enter a point for the COG.

Fan



Create a fan object for fluid dynamics.

On a View window enter three points that define the fan geometry.

Gravity



Create a distant gravity field.

On a View window enter two points that define the direction and force of the distant gravity.

Spin



Set spin of the selected objects.

Select objects to define the spin for. Enter three points that define the spin.

Velocity



Set velocity of the selected objects.

Select objects to define the velocity for. On a View window enter two points that define the velocity vector.

The Velocity tool has the following option:

☐ Selected Points

- Selected Points - Set velocity to selected points only.

Transformation Tools

Available GUI Objects

The Transformation tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the transformation tools.



These tools can be drag&dropped to a tool bar.

Bend



Bend the selected objects. If the object is in point editing mode then the tool will have effect on the selected points only.

Select the objects to be bent. Define the start position for the bend by clicking in a View window. Click to define the end position for bending. The object will be bent between the start and the end points. Move the mouse to get the desired bending and accept the tool by clicking in a View Window.

Extend



Scale the selected objects in the given direction.

Select the objects to be scaled. Enter the center of scaling by clicking in a View Window. Define another point by clicking in a View Window. Move the mouse until the desired effect is achieved and click in a View Window to accept the tool.

The Extend tool has the following options:

☐ About COG

- About COG - Extend object about the center of gravity.

IK



Apply inverse kinematics to the selected points of the selected objects.

Select the object to be modified using Inversed Kinematics. Currently only NURBS curves and Skeletons are supported. Enter the Edit state and select the point/joint to be modified. Click the left mouse button in a View window to start I.K. Move the joint to desired position and click the left mouse button to accept the tool.

The IK tool has the following options:



- Type - I.K. type
- Full I.K. - apply inverse kinematics in both directions
- I.K. head, move tail - apply inverse kinematics only towards the head
- Move head, I.K. tail - apply inverse kinematics only towards the tail
- Local - Cause I.K. to affect the current bone only.

Move



The Move tool can be used to translate the selected objects, handles, lattice coordinates, or texture UV coordinates.

Select the objects to be moved. Specify options in the Control Bar. Click in a View Window to define the start position. Move the mouse until the desired translation is achieved and click again to accept the tool.

The Move tool has the following options:



- Target - Transformation Target.
- Collision Deformation - Activate collision detection system.

Mirror



Mirror the selected objects about given axis.

Select the objects to be mirrored. Click the left mouse button in a View Window to specify the start position of the mirroring axis. Move the mouse until the desired mirroring effect is achieved and click the left mouse button to accept the tool.

The Mirror tool has the following options:

☐ About COG

- About COG - Mirror object about the center of gravity.

Noise



Apply fractal noise deformation to the selected freeform objects. If the object is in point editing mode then the tool will have effect on the selected points only.

Select the objects to be deformed. Define the start position for the noise deformation by clicking in a View window. Click to define the end position for the deformation. The object will be deformed between the start and the end points. Move the mouse to control the level of deformation and accept the tool by clicking in a View Window.

Rotate



The Rotate tool can be used to rotate the selected objects, handles, lattice coordinates, or texture UV coordinates.

Select the objects to be rotated. Specify options in the Control Bar. click in a View Window to define the center of rotation. Click in a View Window to define the point you want to rotate about the rotation center. Move the mouse until desired rotation is achieved and click to accept the tool.

The Rotate tool has the following options:



- Target - Transformation Target.
- Collision Deformation - Activate collision detection system.
- About Pivot - Rotate object about the pivot point.

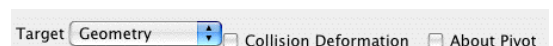
Scale



The Scale tool can be used to scale the selected objects, handles, lattice coordinates, or texture UV coordinates.

Select the objects to be scaled. Specify options in the Control Bar. Enter the center of scaling by clicking in a View Window. Define another point by clicking again. Move the mouse until the desired size is achieved and click to accept the tool.

The Scale tool has the following options:



- Target - Transformation Target.
- Collision Deformation - Activate collision detection system.
- About Pivot - Scale object about the pivot point.

Size



The Size tool can be used to uniformly resize the selected objects, handles, lattice coordinates, or texture UV coordinates.

Select the objects to be resized. Specify options in the Control Bar. Enter the center of scaling by clicking in a View Window. Define another point by clicking again. Move the mouse until the desired size is achieved and click to accept the tool.

The Size tool has the following options:

- Target - Transformation Target
- Deform - Deformation type
- Uniform - No deformation
- Deform in function of distance to a point
- Deform in function of distance to an axis
- Deform in function of projection to an axis
- Apple Curve - Control curve that defines the shape of the deformation.
 - Triangle
 - Gaussian
 - Circle
 - Random
 - Linear
- About Pivot - Size object about the pivot point.

Skew



The Skew tool can be used to skew the selected objects, handles, lattice coordinates, or texture UV coordinates.

Select the objects to be skewed. Specify options in the Control Bar. Select the base point of the skew by clicking in a View Window. Define another point from the View window to define start point for skewing. Move the mouse to skew the object. Click in a View Window to accept the tool.

The Skew tool has the following options:



- Target - Transformation Target.
- Deform - Deformation type.
- Uniform - No deformation
- Deform in function of distance to a point
- Deform in function of distance to an axis
- Deform in function of projection to an axis
- Apple Curve - Control curve that defines the shape of the deformation:

- Triangle
- Gaussian
- Circle
- Random
- Linear
- Collision Deformation - Activate collision detection system.
- About Pivot - Skew object about the pivot point.

Triset Tools

Triset object specific control bar

When a Triset object is selected the control bar shows the following Triset specific tools:

Add Face



Add face to three selected points.

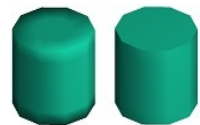
Select exactly three points of a triset object and click the tool button.

Auto Smooth



Adjust surface normals automatically.

This tool examines the true geometric angles between the faces. If an angle between two adjacent faces is lower than a given threshold value, the respective normals are set so that shading appears smooth over the edge between the faces. Otherwise normals are set to match the true geometric normal, and the edge will appear sharp.



In object mode, all normals are processed. The tool also flips all normals to a consistent direction. In point/edge/face editing mode, the tool changes only selected areas of the surface, and keeps the normal direction at the same side of the face as it originally was.

*A cylindrical triset
before and after Auto
Smooth*

Delete



Delete selected points, edges, or faces.

Flat



Flatten normals of the selected faces.

Flip



Inverts the direction of surface normals. In object mode, all normals are flipped. In edit mode, only the selected areas are flipped.

Merge



Merge selected vertices into one vertex.

Select

Selection mode. Select either by vertices, edges, or faces.

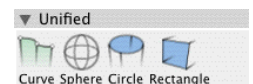
Smooth



Smooth normals of the selected points or faces.

Unified Creation Tools

The Unified tool group in the Tools tab of the Available GUI Objects window (opened by selecting the Environments/Customization/Available Object Window... menu) contains the unified creation tools.

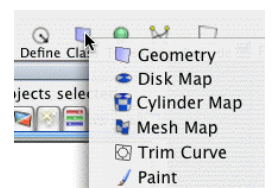


These tools can be drag&dropped to a tool bar.

Circle



With the Circle tool you can create objects with a circular cross section. I.e. circles, disks, cylinders, cones, cut cones, hyperboloids and ellipsoid segments. Also NURBS meshes and Subdivision Surface objects of those shapes can be created. Disk Map, Cylinder Map, Mesh Map objects and circular trim curves can also be created. The tool can also be used for 3D painting circular curves to underlying objects that have bitmap textures.



The Circle tool has the following options:

Define - How the circle is defined. The choices are:

- Center & Radius
- Center & Axes
- Corner & Corner
- Three points in arc

- Center & Corner

Class - Class of the circular object to be created.

- **Geometry** - Creates a geometry object.

Type - type of the geometry object



- *Analytic*



- Extrude
 - Planar - Creates circles and disks.
 - Extruded - Creates cylinders, cones and cut cones.
 - Hyperboloid - Creates hyperboloids and ellipsoid segments.
- Filled - If checked, closes the caps
- Arc - If checked creates open arcs

- *Nurbs*



- Order - order of the NURBS curve or surface
 - Polygonal
 - Quadric
 - Cubic
- Extrude
 - Planar - Creates planar circular NURBS curves or NURBS disks.
 - Extruded - Creates NURBS meshes shaped as cylinders, cones and cut cones.
 - Hyperboloid - Creates NURBS meshes shaped as hyperboloids and ellipsoid segments.
- Filled - If checked, closes the caps.
- Arc - If checked creates open arcs.
- Exact - If checked creates an exact NURBS circle.
- Points - Number of points in curve or surface.
- Density - Number of extra curves

- *Subdivision*



- Extrude
 - Planar - Creates planar circular SDS disks.
 - Extruded - Creates SDS objects shaped as cylinders, cones and cut cones.

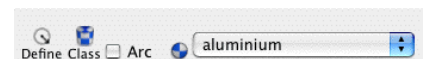
- Hyperboloid - Creates SDS objects shaped as hyperboloids and ellipsoid segments.
- Arc - If checked creates open arcs
- Points - Number of points in curve or surface
- Density - Number of extra curves
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

- **Disk Map** - Creates a disk mapping.



- Arc - If checked creates open arcs.
- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped

- **Cylinder Map** - Creates a cylinder mapping.



- Arc - If checked creates open arcs
- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped

- **Mesh Map** - Creates a mesh mapping.



- Order - order of the NURBS curve or surface
 - Polygonal
 - Quadric
 - Cubic
- Filled - If checked, closes the caps
- Arc - If checked creates open arcs

- Points - Number of points in curve or surface
- Density - Number of extra curves
- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped
- **Trim Curve** - Creates a circular trim curve.



- Order - order of the NURBS trim curve
 - Polygonal
 - Quadric
 - Cubic
- Filled - If checked, closes the trim curve
- Arc - If checked creates an arc
- Points - Number of points in the trim curve

- **Paint** - Paints circular curves to underlying objects that have bitmap textures.

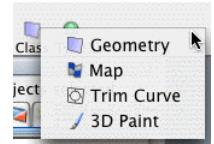


- Order - order of the NURBS curve
 - Polygonal
 - Quadric
 - Cubic
- U: - Number of points in the curve.
- Line Width - width of the circular curve to be painted.
- Filled - Select either a circle or a filled circle (i.e. disk).
- Color - color of the circle or disk to be painted.

Curve



With the Curve tool you can create freeform NURBS curves, trim curves, NURBS surfaces and extruded NURBS objects with freeform intersection, polygons, polyhedrons, polymids, and mesh map objects. The tool can also be used for 3D painting freeform curves to underlying objects that have bitmap textures.



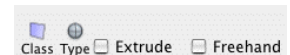
The first option of the curve tool, Class, let's you choose what kind of objects to create.

Geometry - Creates an analytic polygon, polymid, polyhedron, or a NURBS curve or a NURBS mesh.



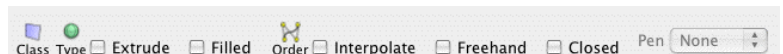
Type:

- *Analytic* - create polygon, polyhedron or polymid



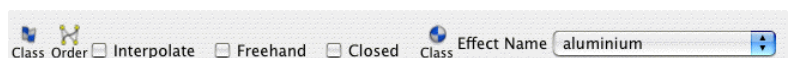
- Extrude - If set, creates extruded objects, i.e. polyhedrons and polymids.
- Depth - Depth of the extruded object.
- Freehand - If set, the polygon, polyhedron, or polymid is defined by moving the mouse while holding down the left mouse button.

- *Nurbs* - create NURBS curve or mesh



- Extrude - If set, creates extruded objects.
- Filled - If set, creates a surface whose boundary is the defined freeform curve
- Order - order of the NURBS curve
 - Polygonal
 - Quadric
 - Cubic
- Interpolate - If set, the curve passes through the defined points. Available only when a Cubic curve is created.
- Freehand - If set, the curve is defined by moving the mouse while holding down the left mouse button.
- Closed - If set, defines a closed curve or surface.
- Pen - Create a NURBS surface using an existing, selected curve as a pen.
 - None - creates a curve, no surface.
 - Sweep - the selected 'pen' curve is swept along the curve drawn and rotated according to the curvature of the curve drawn.
 - Rail - the selected 'pen' curve is swept along the curve drawn. The pen object is not rotated while sweeping.
 - Rotate - the pen object is used as an axis and the new curve is rotated around it.

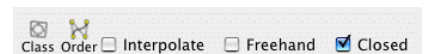
Map - Creates a mesh mapping.



- Order - order of the NURBS curve
 - Polygonal

- Quadric
- Cubic
- Interpolate - If set, the curve passes through the defined points. Available only when a Cubic curve is created.
- Freehand - If set, the curve is defined by moving the mouse while holding down the left mouse button.
- Closed - If set, defines a closed curve or surface.
- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped

Trim Curve - Creates a trim curve. A NURBS mesh must be selected.



- Order - order of the NURBS curve
 - Polygonal
 - Quadric
 - Cubic
- Interpolate - If set, the curve passes through the defined points. Available only when a Cubic curve is created.
- Freehand - If set, the curve is defined by moving the mouse while holding down the left mouse button.
- Closed - If set, defines a closed curve.

3D Paint - Paints curves to underlying objects that have bitmap textures.

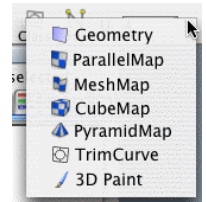


- Order - order of the NURBS curve
 - Polygonal
 - Quadric
 - Cubic
- Interpolate - If set, the curve passes through the defined points. Available only when a Cubic curve is created.
- Freehand - If set, the curve is defined by moving the mouse while holding down the left mouse button.
- Closed - If set, defines a closed curve.
- Line Width - width of the curve to be painted.
- Filled - Select either filled curve or non-filled.
- Color - color of the curve to be painted.

Rectangle



With the Rectangle tool you can create objects with a rectangular cross section. I.e. rectangles, cubes, pyramids and cut pyramids. Also NURBS meshes and Subdivision Surface objects of those shapes can be created. Parallel Map, Mesh Map, Cube Map, Pyramid Map objects and rectangular trim curves can also be created. The tool can also be used for 3D painting rectangular shapes to underlying objects that have bitmap textures.



The Rectangle tool has the following options:

Define - How the rectangle is defined. The choices are:

- Corner - Corner
- Center - Corner

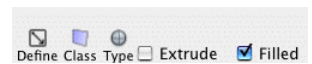
Class - Class of the rectangular object to be created.

- **Geometry** - Creates a geometry object.

Type - type of the geometry object

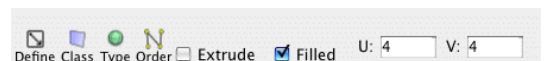


- *Analytic*



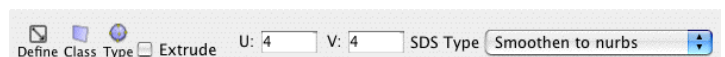
- Extrude - If checked creates extruded objects, i.e. Cubes, pyramids, cut pyramids.
- Filled - If checked, closes the caps.
- Center - If checked centers the apex. Available only when creating extruded objects

- *Nurbs*



- Order - order of the NURBS curve or surface
 - Polygonal
 - Quadric
 - Cubic
- Extrude - If checked creates NURBS mesh objects shaped as Cubes, pyramids, cut pyramids.
- Filled - If checked, closes the caps
- Center - If checked centers the apex. Available only when creating extruded objects.
- U - Number of points in the width (u) direction.
- V - Number of points in the height (v) direction

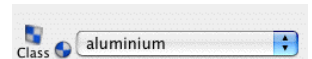
- *Subdivision*



- Extrude - If checked creates SDS objects shaped as Cubes, pyramids, cut pyramids.

- Filled - If checked, closes the caps
- Center - If checked centers the apex. Available only when creating extruded objects
- U - Number of points in the width (u) direction.
- V - Number of points in the height (v) direction.
- W - Number of points in the depth (w) direction.
- SDS Type - SDS object type. Possible values are:
 - Smoothen to nurbs
 - Smoothen to rectangles
 - Smoothen to triangles
 - Polygonal

- **Parallel Map** - Creates a parallel mapping.



- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped

- **Mesh Map** - Creates a mesh mapping.



- Order - order of the NURBS surface
 - Polygonal
 - Quadric
 - Cubic
- U - Number of points in the width (u) direction
- V - Number of points in the height (v) direction
- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped

- **Cube Map** - Creates a cube mapping.



- Class - class of the map created

- Material - Map a material
- Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped.
- Environment Map - check this to create an environment map.
- **Pyramid Map** - Creates a pyramid mapping.



- Class - class of the map created
 - Material - Map a material
 - Effect - Map a particle post effect
- Effect Name - name of the material or particle post effect to be mapped
- **Trim Curve** - Creates a rectangular trim curve.



- Order - order of the NURBS trim curve
 - Polygonal
 - Quadric
 - Cubic
- Filled - If checked, closes the trim curve
- U: - Number of points in the trim curve

- **Paint** - Paints rectangular shapes to underlying objects that have bitmap textures.



- Order - order of the NURBS curve
 - Polygonal
 - Quadric
 - Cubic
- U: - Number of points in the painted line.
- Line Width - width of the rectangular line to be painted.
- Filled - Select either a filled or non-filled rectangle.
- Color - color of the rectangle to be painted.

Sphere



With the Sphere tool you can create spherical NURBS and Subdivision Surface objects, analytic spheres, and sphere mapping objects.

The Sphere tool has the following options:



Input - How the sphere is defined, by center and radius, two corners, or center and corner.

Class - class of the sphere created.

- *Geometry*

Type - Type of the geometry



- Analytic
- Nurbs
- Subdivision

- *Sphere Map*



- Map - Type of class to map
 - Material - Map a particle post effect
 - Effect - Map a post particle effect
- Effect Name - name of the material or particle post effect to be mapped.
- Environment Map - check this to create an environment map.

User Interface

Geometry Managers

The geometry managers can be Drag&dropped to empty Drop Target windows.

To learn more about using the geometry managers in GUI editing see the User Interface / Customizing the User Interface chapter in the manual.

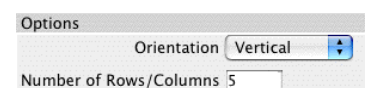
Placer

- The placer can be used when you want to position the child windows or gadgets positions relative to the width and height of the window.

RowCol

- The RowCol can be used when you want to lay the child windows or gadgets in multiple rows or columns.

The RowCol has the following options:



Orientation - Orientation of the RowCol (Vertical/Horizontal).

Number of Rows/Columns - Number of rows or columns in the RowCol

Packer



The packer can be used when you want to organize the children either vertically or horizontally.

The packer has the following options:

Orientation - Orientation of the packer, Vertical or Horizontal.



Macro GUI gadgets

The macro GUI gadgets can be inserted to the GUI by dragging from the buttons and dropping them to the GUI.

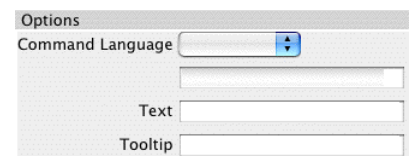
Macro Slider



The macro slider is a slider with an associated script. When the slider value is changed the script is executed.

The macro slider has the following options:

- **Command Language** - Language of the script.
- **Script** - Script to be executed when the slider value changes.
- **Text** - Label text for the slider.
- **Tooltip** - Tooltip for the slider.



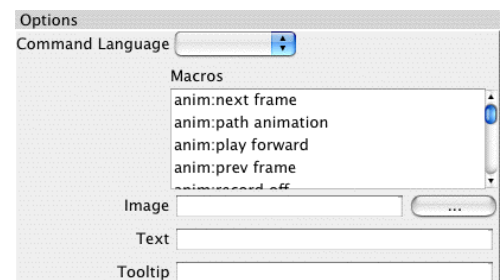
Macro Button



The macro button executes either a macro or a script when clicked.

The macro button has the following options:

- **Command Language** - Language of the script. Leave empty to select a macro instead of writing a script.
- **Script** - Macro to be executed when the button is clicked.
- **Macros** - Displays the list of available macros when no script language is selected. You can select the target macro from the list.
- **Image** - Image file to show on the macro button.
- **Text** - Label text for the macro button.
- **Tooltip** - Tooltip for the macro button.



Windows

The windows can be inserted to the GUI by dragging from the buttons and dropping them to the GUI. See the User Interface / Customizing the User Interface chapter in the manual to learn more about GUI editing.

Color Square

 Color Square gadget for creating predefined colors.

The Color Square gadget is a rectangle that shows a constant color. You can drag&drop the color from a Color Square gadget to a View window to set the color of the currently selected objects. The color can also be drag&dropped to any color gadget in e.g. property gadgets or tool gadgets. The color of the Color Square gadget can be set by double-clicking the gadget and specifying the color using the color dialog that opens.

Color

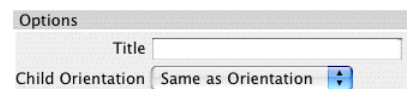
 The color gadget allows to define a color numerically. You can drag&drop the color from the color canvas of a Color gadget to a View window to set the color of the currently selected objects. The color can also be drag&dropped to any color gadget in e.g. property gadgets or tool gadgets.

Client

 Client windows are floating windows that can be docked to container windows.

The client window has the following options:

Title - Title for the window.



Child Orientation - Select whether and how the children's orientation follow the Client window's orientation.

- Same as Orientation - The children will have the same orientation as the Client window.
- Opposite - The children's orientation will be opposite to that of the Client window.
- Original - The children retain their native orientation.

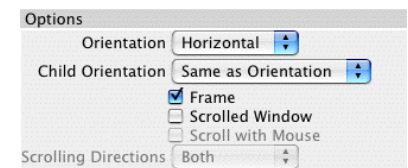
Container Window

 Container window is where Client windows can be docked.

The container window has the following options:

Orientation - Orientation of the container window. Vertical or Horizontal.

Child Orientation - Select how the child Client's orientation follows the Container window's orientation.



- Same as Orientation - The child Client will have the same orientation as the Container window
- Opposite - The child Client's orientation will be opposite to that of the Container window

Frame - Check to create a thin frame around the container window.

Scrolled Window - Check to make a window whose contents can be scrolled if they do not fit inside the window.

Scroll with Mouse - Check this if you want to scroll with the mouse.

Scrolling Directions - Select in which directions the window will scroll.

- Both

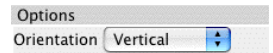
- Horizontal
- Vertical

Separator

 Use this gadget to separate other windows.

The separator gadget has the following options:

Orientation - Orientation of the separator gadget, Horizontal or Vertical.



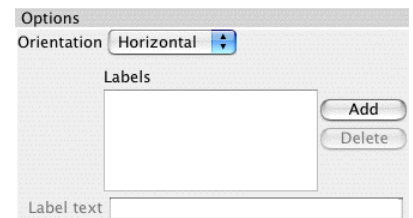
Cycle

 The cycle gadget has a drop down list gadget where different 'pages' can be selected. Other gadgets can be drag&dropped to the pages. Only one page at a time is visible.

The cycle gadget has the following options:

Orientation - Orientation of the cycle gadget, Vertical or Horizontal.

Labels - Shows the labels that are to be created in the cycle.



Add - Add another label.

Delete - Delete the selected label. Select the label in the Labels gadget.

Label text - Change the selected label. Select the label in the Labels gadget.

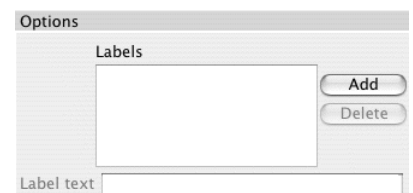
Tab

 The tab gadget has one or more labeled tabs. Selecting a tab opens a different page. Other gadgets can be drag&dropped to the tab pages. Only one page at a time is visible.

The tab gadget has the following options:

Labels - Shows the labels that are to be created in the tab control.

Add - Add another label.



Delete - Delete the selected label. Select the label in the Labels gadget.

Label text - Change the selected label. Select the label in the Labels gadget.

Frame

 Creates a frame with text. Other gadgets can be drag&dropped inside the frame.

The frame gadget has the following options:

Text - Text label on the frame.



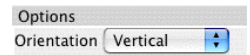
Checkbox - Check this if you want to create a frame that can be closed.

Splitter

 Splitter is a gadget that divides its parent window either vertically or horizontally. The splitter can be dragged and moved with the mouse. Other gadgets can be drag&dropped to both sides of the splitter gadget.

The splitter gadget has the following options:

Orientation - Orientation of the splitter gadget. Vertical or Horizontal.



Window

 Drop target window

The drop target window has the following options:

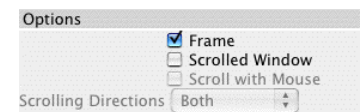
Frame - Check to create a thin frame around the drop target window.

Scrolled Window - Check to make a window whose contents can be scrolled if they do not fit inside the window.

Scroll with Mouse - Check this if you want to scroll with the mouse.

Scrolling Directions - Select in which directions the window will scroll.

- Both
- Horizontal
- Vertical



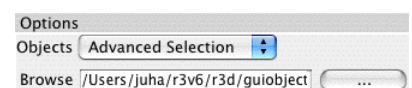
Load

 Load a saved GUI object as part of the GUI.

The Load tool has the following options:

Objects - Select a GUI object to load.

Browse - Browse for a file containing a saved GUI object.



Float

 Open a floating window.

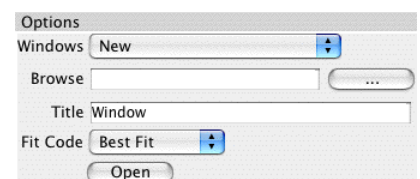
The Float tool has the following options:

Windows - Select a predefined window or create a new one.

Browse - Browse for a file containing a saved floating window.

Title - Title for the floating window.

Open - Click here to open the floating window.



Windows

Advanced Selection Window

The Advanced Selection Window enables geometric object selection by object properties. For example, all material mapping objects hidden in a complex hierarchy can be selected here easily and made invisible to avoid wire frame clutter.

The window has three tabs for three different selection approaches: by name, by object class and by other object property. The three buttons on the bottom perform the actual selection operation after the condition has been defined:

- **Clear** deselects all objects providing a clean start for a new selection
- **Select** adds the objects that fulfill the given condition to the current selection
- **Deselect** removes the objects that fulfill the given condition from the current selection



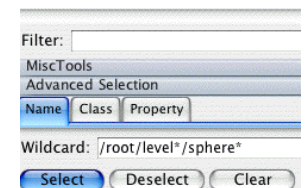
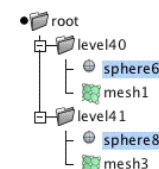
Note

Select operations do not perform a completely new selection but add and subtract from an already existing selection. This way you can combine several consecutive select operations.

Select operations search through the entire object tree (not the current Level only).

Only the condition set by the visible tab (Name/Class/Property) affects selection. Hidden conditions are not considered.

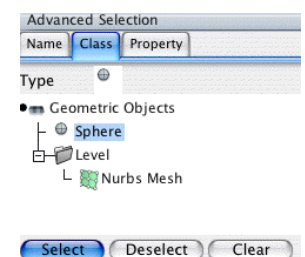
The string that specifies object selection should include full hierarchy path to the desired objects, for example: '/root/level*/sphere*'. The wildcard character asterisk '*' matches any number of consecutive characters.



Selection by name wildcards

Select By Class

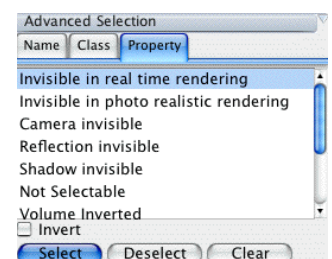
The 'Class' tab of the advanced selection window shows all object types included in the current project. You can pick for example 'Sphere' class and hit 'Select' to get all analytic spheres selected.



Selection by object class

Select By Property

The 'Property' tab shows all common on/off type attributes of geometric objects. Select adds objects, which have the activated property set. If 'Invert' is checked, select adds objects, which do not have the property set.



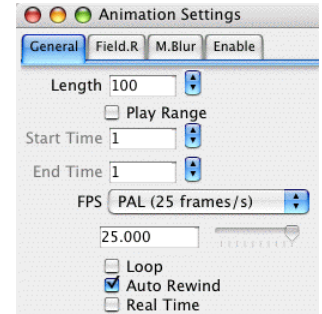
Selecting all 'wire frame invisible' objects

Animation Settings

The animation settings window is available as a 'docked' interface item below the select window and in the Windows pull down menu - the latter opens a floating window containing the animation controls.

The following animation options are available:

- *Length*: The length of animation measured using the current time system (Frames/Real time/Seconds)
- *Field Rendering*: Activates field rendering. Field rendering means the method of rendering animation frames, which consist of two interlaced 'fields'. Fields split the time between frames to half. The result is smoother animation. Common video and broadcasting standards (PAL, NTSC, VHS, etc.) use fields.
- *Shift latter field up*: By default, the lines of the second field (which comes later in time) of a frame are computed below the lines of the first field. If this option is checked, it inverts the order.
- *Interlace latter field above*: By default, the lines of the second field are placed under the corresponding lines of the first field, when fields are combined into a single frame by interlacing them. The frame lines will then be as follows: line1 of field1, line1 of field2, line2 of field1, line2 of field2, etc. This option starts interlacing from the latter field: line1 of field2, line1 of field1, line2 of field2, line2 of field1, etc. This is known as starting with the "even" field (by default) or starting with the "odd" field. This setting is dependent on the output device you are using.
- *Motion Blur*: Turns motion blur on. Every image of the animation (field or frame) will be assembled from the given amount of sub images distributed evenly between the fields/frames. This will make motions appear smoother.
- *Samples*: The amount of sub images for motion blur. The higher the amount, the better the quality, but rendering time increases respectively
- *Loop*: If activated, animation playback rewinds and restarts automatically. This can be useful when previewing cyclic animations. Note that when using File Renderer, this option is ignored
- *Real Time*: If activated, animation playback speed is determined by the FPS (Frames Per Second) field. The system either skips frames, or delays rendering to synchronize the play back speed to the given FPS value. Note that simulations may not work correctly with this option.
- *Auto Rewind*: If activated, animation playback rewinds automatically at the end. When converting simulations to keyframe animations, auto rewinding can be turned off:
 - Turn auto rewind off
 - Set animation recording on
 - Play the simulation
 - After playback, select the root level and apply Copy tool to it
 - Rewind the animation
 - Delete the restored simulation hierarchy and paste the keyframe animation to replace the simulation.
 - Turn off Simulation for all simulated scene parts. In the choreography editor, remove all unnecessary animated attributes (velocity, spin etc). Apply the Simplify curve tool to the remaining keyframe curves as many times as necessary.
 - Turn auto rewind on again



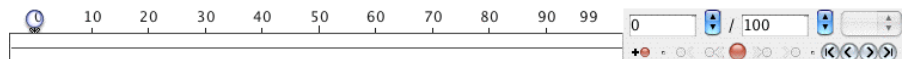
- **Enable Simulations:** If set, physical simulations (gravity, collisions etc.) are computed during playback.
- **Animate Materials:** If set, material attribute changes can be keyframed by changing their value animation recording on. The option also controls if already keyframed material changes will be assigned during the playback.
- **Animate Image Effects:** If set, image effect attribute changes can be keyframed by changing their value animation recording on. The option also controls if already keyframed changes will be assigned during the playback.
- **Animate Particle Effects:** If set, particle effect attribute changes can be keyframed by changing their value animation recording on. The option also controls if already keyframed changes will be assigned during the playback.
- **Animate Images:** If set, image attribute changes can be keyframed by changing their value animation recording on. The option also controls if already keyframed changes will be assigned during the playback.
- **Play Range:** If set, the playback steps through the given frame sub interval instead of the whole animation
- **Start:** The beginning of the 'Play Range' interval
- **End:** The end of the 'Play Range' interval
- **FPS:** Defines how many frames one second of animation contains. The common standards are available as presets.

Animation Window

The animation window is available in the default user interface at the bottom of the screen. It can be hidden and made visible using the pull down menu 'View/Animation Window' or by hitting F7 shortcut key.

The time line of the animation window and other time related controls follow the selected time display conventions, which can be specified in Options/Window/Animation. The window is synchronized with the Select Window showing the keyframes of the currently selected object / choreography. For example, to edit key frames of a material object, click the Material tab of the select window, select the desired material and then select the desired keyframer choreography

The following controls are available:



Play range start: Defines the beginning of the animation playback range. The control is visible only when the Play Range option is enabled. Dragging the triangular marker with the left mouse button changes the play range start.

10

Play range end: Defines the end of the animation playback range.

30

Time slider: Shows the current time of the animation on the timeline. Dragging the knob with the left mouse button changes the current time.

60 70

LMB clicking the time line around the knob changes the time one frame to the respective direction. RMB click on the time line moves the knob to the clicked position. Also the following modifiers are supported:

- Shift + RMB click moves the knob to the nearest key frame.
- Alt + RMB click plays the knob to the nearest key frame.

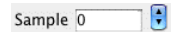
Current: Shows the current frame of the animation. Entering a new frame number to this field jumps (not plays) the animation time to the new frame.

42

Field: Shows the current field (odd/even). The field control is visible when Field Rendering is enabled. The current field can be changed using the gadget.



Sample: Shows the current motion blur sample. Entering a new index to the gadget changes the current sample. This gadget is visible when Motion Blur is enabled.



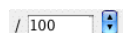
Playback controls: Rewinds the animation back to the first frame (or to the beginning of the play range if the range option enabled). Plays the animation (or a defined sub range) backwards or forwards and jumps to the last frame.



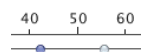
Record: Turns animation recording on/off. When recording is enabled, all object modifications become recorded into the current choreography. If the current choreography is not a key framer, one is automatically created.



Total: Controls the amount of frames in the animation. If the frame count is changed and the current project contains animated objects, the program asks if the time lines should be scaled respectively.

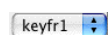


Keyframes: Shows the keyframes of the currently selected objects. Key frames can be edited by dragging them with the mouse. The keyframe gadget can be zoomed and panned using the standard Shift and Apple modifiers while dragging with RMB.



- Attributes - Sub menus can be used for enabling / disabling attributes for key framing. If an attribute is disabled, editing attributes with animation recording on will not create new key frames for it.
- Show Choreographs, Hide Choreographs - These menus can be used for controlling the visibility of choreographs in the Choreography Window. Useful when managing scenes consisting of hundreds or thousands of animated objects.
- Up, Down - These menus can be used for controlling the current choreography. Only choreographs up to the current choreography are executed.
- Reset - Animation time line can be zoomed and panned via RMB + Apple and Shift keys. This menu restores the default scale and pan values.
- Fix attribute list - This menu fixes the current animated attribute list so that no new attributes will be animated. However, the attributes already animated will be treated as usual.

Choreography stack: Shows the choreography stack and the current choreography.



Choreography stack contains all the choreographs applied to the object. The first choreography (typically named 'init') defines the initial state of the attribute in question. Any number of choreographs can be stacked on top of each other. Keyframer choreography can be used for animating attributes. It is automatically created when the user modifies objects when animation recording is activated. Pose choreography is typically created when the user modifies animated objects with animation recording off.

Create New Key: Creates new keyframes to the current choreography. This tool is available only when the type of the current choreography is 'Keyframer'. Another way of creating keyframes is by editing objects with animation recording on.



Previous Key: Moves the current time knob to the previous key frame.



Next Key: Moves the current time knob to the next key frame.



Play to previous key: Plays animation to the previous key frame.



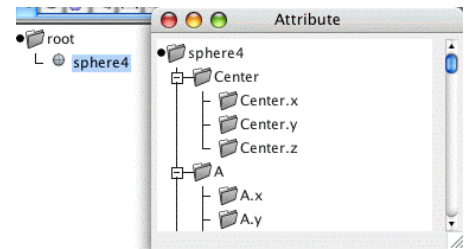
Play to previous key: Plays animation to the next key frame.



Attributes Window

The attributes window shows the list of an object's public attributes. The object whose attributes should be displayed can be selected by dragging and dropping it from the select window to the attributes window. All object types (not only geometric objects, but also image effects etc.) are valid for this purpose.

The displayed attributes can be drag and dropped into the scripting window. This helps in avoiding typing errors.



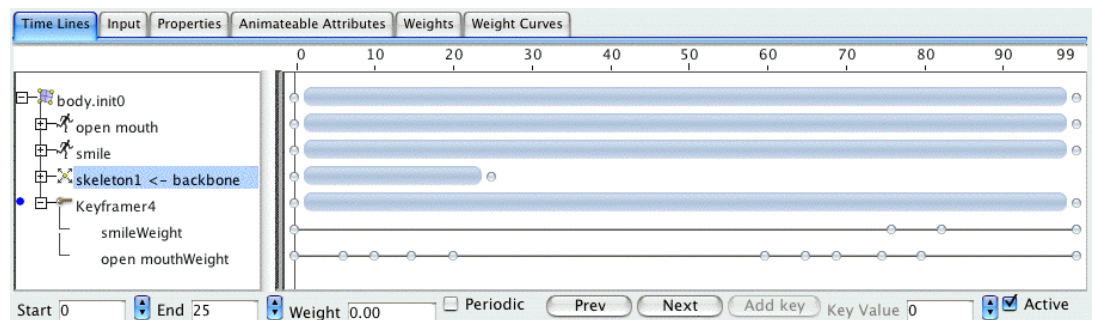
Choreography Window

The choreography window contains the tools for editing animation timing and other animation related features. The standard program configuration includes a floating choreography window which can be opened from the 'Animation/Choreographs..' pull down menu or by hitting F6 hotkey.

The window consists of several tabs, which are listed below.

Time Lines

The Time Lines tab consists of a choreography list, a time line table and a number of controls on the bottom of the window. Moving the splitter line that divides the window vertically controls the space division between the choreography list and the time lines.



The following controls are available:

Choreography list: Shows all choreographs of the current project (or alternatively only the choreographs of the selected objects - see the Only Selected Objects option below). Each animated object has an initialization choreography, which defines its state in the beginning of the animation. An initialization choreography can have any number of sub choreographs, which are executed in the order they appear in the choreography list. A LMB click activates a choreography, after which its properties can be modified using the controls on the bottom of the window, with popup menus and with other tabs of the window. The same list is visible also in many other tabs of the choreography window.

A double click on a choreography with the left mouse button opens the sub structure of the choreography, showing the sub choreographs or the animated attributes (and their time lines on the right side) included in the choreography.

A second double click hides the sub structure. A complex animation often contain very many choreographs. Keeping only choreographs under construction opened helps in managing the complex time line system.

A slow double click activates the rename tool. Assigning descriptive names for choreographs helps to manage complex animations. For example, facial expression choreographs can be renamed as 'smile', 'cry', 'anger' etc.

The choreography list has the following popup menu:

- *New/Keyframer, ...*: Creates a new sub choreography for the target object of the selected choreography. The time line is initialized to fill the parent time line.
- *Duplicate*: Duplicates the selected choreography. Initialization choreographs cannot be duplicated.
- *Split*: Uses the current time to cut the time line of the selected choreography into two parts. The result is two key frame choreographs where the original defines motions before the cut and the newly created choreography defines the motions after the cut point. The actual motions remain unchanged. If the tool is applied to the root choreography, all keyframe sub choreographs are split.
- *Delete*: Deletes the selected choreography. If the deleted choreography is the initialization choreography, the object becomes unanimated.
- *Rename*: Activates the rename tool (slow double-click does the same).
- *Open*: Open the sub structure of the selected choreography.
- *Close*: Close the sub structure of the selected choreography.
- *Open All*: Opens the sub structure of all choreographs.
- *Close All*: Hides the sub structure of all choreographs.
- *Only Selected Objects*: If set, the usual object selection at the select window controls which choreographs are shown. This helps to control and edit complex animations.
- *Load*: Loads a choreography from disk and assigns it to the target object of the selected choreography.
- *Save*: Saves the selected choreography to a disk file.

Time lines: The time line gadget shows a time scale, a slider for changing the current time and a table of horizontal time lines. Dragging the knob on the time scale with the left mouse button changes the current time. A RMB click moves the time to the clicked position.

A time line can be edited by dragging the end knobs with the left mouse button, or by using the numeric 'Start' and 'End' controls on the bottom of the window. The time line can be moved by dragging it.

By default changing the lenght of a time line scales also the associated key frames. This can be avoided by holding down the Alt key, in which case the associated keys are scaled inversely so that their positions in global time remains unchanged.

If the sub structure of a keyframer choreography is opened, the time line table shows also the attribute specific animation keys. Animation keys are shown as rectangular knobs. Keys can be activated by LMB clicking them. Shift modifier allows key multi-selection. LMB-dragging can also be used for multi-selection. Keys can be moved by dragging them with the mouse. Selected keys can be moved also using the 'Key Value' gadget below the time line table. The Shift modifier actives frame grid when knots are being moved.

Keys can be duplicated by draggning the selected keys while holding down the Apple key.

For accurate editing, you can zoom in by dragging with the right mouse button Shift key held down. Apple-RMB-dragging pans the time lines (i.e. selects the time range to be displayed on the window). Mouse wheel can be used for scrolling the window.

The time line area has the following popup menu:

- **Delete Key Values:** Deletes a selected key.
- **Delete All Keys:** Deletes all keys of attributes, which have one or more keys selected. Those attributes become unanimated.
- **Reset:** Resets zooming and positioning to show the full animation length.

Start: The start time of the selected choreography (in frames or real time depending on Options/Animation/Time System).

End: The end time of the selected choreography.

Weight: The weight of the selected choreography in the current frame.

Periodic: Repeats the selected choreography outside the actual time line.

Prev: Jumps the time to the previous key frame. Only keys shown on the time line table are considered.

Next: Jumps the time to the next key frame. Only keys shown on the time line table are considered.

Add key: Adds a new key frame at the current animation frame to the selected attribute or to all attributes of the current choreography.

Key value: The time of the selected key(s). If multiple keys are selected and their values are different, a red color indicates value conflict.

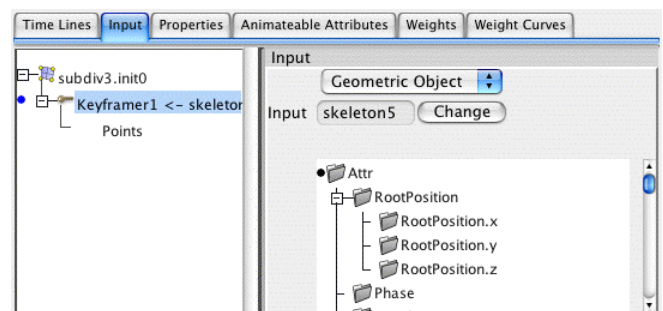
Active: Activates the selected choreography.

Input

The Input tab controls choreography parameterization. By default, choreographs are parameterized by the usual animation time, but here you can set up other kind of dependencies as well. For example, thickness of a muscle can depend on a skeleton angle.

The input parameter can be changed as follows:

By selecting an object from the select window, then choosing the corresponding object type from the selector in the 'Input' area and pressing the 'Change' button. After that, the choreography window displays the attributes of the chosen object and you can activate a suitable parameter for the choreography. For example:



- Animate the desired property (for example, the scale of a muscle).
- Go to the Input tab of the choreography window. Select the keyframer object which animates the property.
- In the select window, select the object, which should control the animated property. It can be another geometry object, a material etc.
- In the choreography window, set the input object type gadget to match the object selection you did. If you selected a material, set Input=Material here, too.
- Hit the Change button.

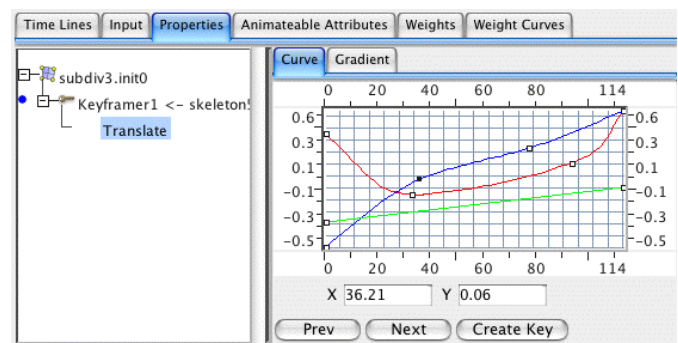
- Select a suitable attribute from the displayed attribute list.

To summarize, the input control gadgets are:

- **Input type selector:** Selects the type of the input object for 'Change' operation. The available types are:
 - *Time*
 - *Geometric Object*
 - *Material*
 - *Particle Effect*
 - *Image Effect*
- **Change:** Changes the input object as specified by the input type selector and object selection.
- **Attribute list:** Shows the list of attributes of the chosen input object. The input parameter can be selected here.

Properties

The Properties tab shows the properties of the currently selected choreography. If a keyframe choreography is selected, the tab shows the actual animation curves, which define the animated attribute values. The attribute for editing can be selected by opening the sub structure of a choreography and clicking an attribute. The following keyframe specific controls are available in this tab:

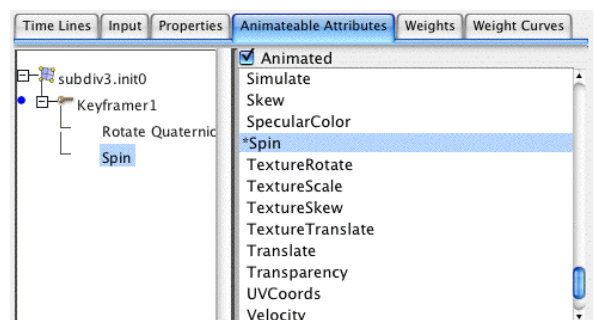


- **Curve Gadget:** A standard curve gadget for editing the selected animation curve.
- **Prev:** Jumps the time to the previous key frame of the animation curve (other hidden attributes do not count).
- **Next:** Jumps the time to the next key frame of the animation curve
- **Create Key:** Creates a key frame to the currently edited animation curves at the current animation time.

For more information about other type of choreographs, consult Choreography Classes reference manual page.

Animateable Attributes

This tab shows the list of all animateable attributes of the target object of the selected choreography. A new attribute can be selected from the list and made animateable using the 'Animated' checkbox above the attribute list. After that, the animation curve can be initialized at the Properties tab using the predefined curve types and edited to a suitable form. Clearing the 'Animated' checkbox for an attribute deletes the corresponding animation curve.





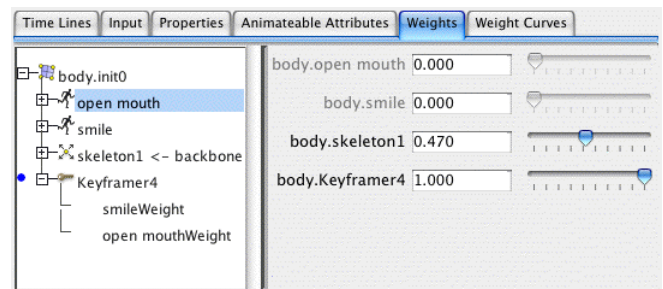
Note

The program automatically adds the attributes of the existing choreographs into a new choreograph.

Weights

The Weights tab contains a choreography weight slider for each choreograph of the current project. Decreasing the weight of a choreography from the full level 1.0 reduces the effect of the choreography. At zero level, the choreography no longer animates the target object.

Choreography weights can be animated as well. You can achieve this by creating a new keyframer choreography and adding the desired weight attributes to it in the Animateable attributes tab. Then you can edit the weight curves of the Weight Curves tab. You can also change time and weight values while animation recording is turned on.



Weight Curves

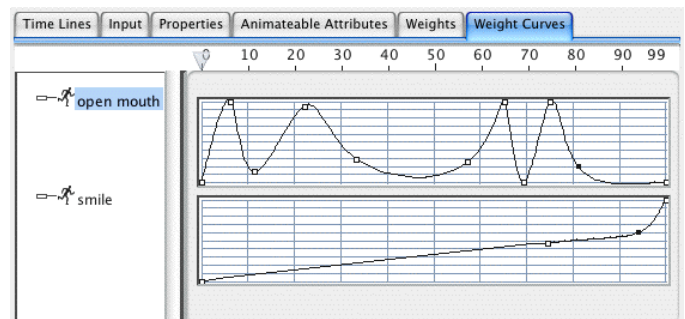
The Weight Curves tab shows the animation curves, which control the weight of choreographs.



Note

A choreography may not have a weight curve. You can make the weight animateable as follows:

- Create a keyframer object to the end of the choreography list of the animated object
- Edit the weight sliders with animation recording mode enabled.



The left side of the 'Weight Curves' tab has the same popup menu tools as the time lines tab, plus some additional menu items for controlling curve visibility:

- *Show Curve*: Shows the weight curve of the selected choreography.
- *Hide Curve*: Hides the weight curve of the selected choreography.
- *Show All Curves*: Shows the weight curves of all choreographs.
- *Hide All Curves*: Hides the weight curves of all choreographs.

The actual curves are displayed using the standard curve gadget. The curves can be edited here.

You can also set weight curves up manually, as follows:

- Create a keyframer object to the end of the choreography list of the animated object

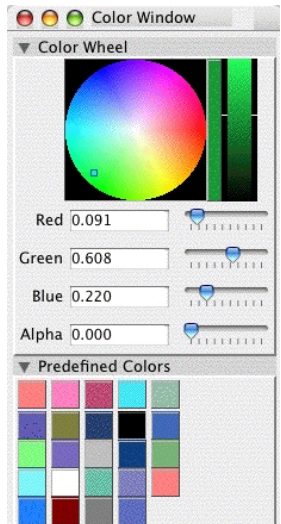
- Add the weight parameter into it using the 'Animateable Attributes' tab
- Use the popup menu of the curve gadget to initialize a curve for further editing.

Color Window

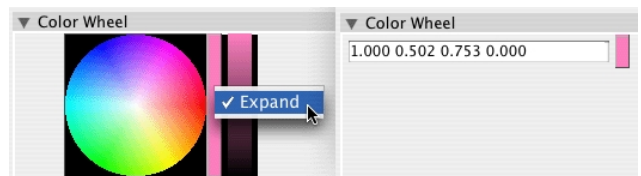
The Color Window is available at the 'Windows' pull down menu. The hotkey F12 of the default key map opens and closes the window. The color gadget of the color window is used as a component in many property sheets, tool controls etc. in a reduced form (without 'Predefined Colors' array). The documentation below is valid for all such color gadgets.

The floating color window opened from the 'Windows' pull down menu does not directly change any color attribute in the scene. It acts as a tool where you can blend colors and optionally store them to the 'Predefined Colors' array. The actual use of created colors happens by dragging and dropping colors from the color window to color gadgets of object property windows, to tool controls etc. Colors can be fetched into the color window the same way, using drag and drop.

The following controls are available:



- **Color Wheel:** You can pick a suitable base color from the circular color wheel gadget. The high, narrow color sample rectangle immediately at the right side of the wheel displays the result color while you LMB drag the mouse on the wheel. The slider at the right adjusts the darkness of the selected color. The color wheel acts as a drop target: dropping a color from another color gadget onto it sets the current color.
- **Current Color Area:** This rectangular area displays the result color defined using other controls of the window. Despite of its small size, it provides some useful functionality:
 - A LMB click opens the standard color dialog of the operating system. You can use it as an alternative way to define the color.
 - A single popup menu 'Expand' is available on it. If cleared, the color wheel and related controls collapse into a minimal state where only a single numeric rgb(a) field and the current color area visible.
 - You can drop colors into this area and drag the color from it into other color gadgets.
- **Brightness Bar:** The vertical bar on the right adjusts the brightness of the selected color. The scale starts from black and ends to the color specified using the wheel. LMB click or drag the bar to change the brightness.
- **Red, Green, Blue, Alpha:** The numeric components of the current color. In the expanded state, the values can be controlled using separate sliders. In collapsed state, a single field defines values, sub components being separated with empty spaces. The alpha channel is optional and not available in all contexts. Some tools, which need the alpha value, show the gadget with the alpha control. For example, the 3D paint airbrush uses the alpha value to define paint transparency.
- **Predefined Colors:** The array of predefined colors is not available in all color gadgets. When available, you can drag colors from it and drop them into other color gadgets. You can also store colors to this array using drag and drop (for example from the color sample area of the color wheel).



The Color Wheel in expanded and collapsed states

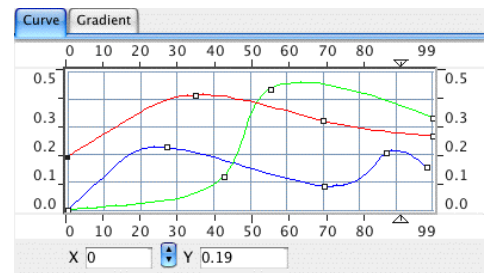
Curve Editor

Two-dimensional curves are used for many purposes in Realsoft 3D: as a VSL shading object, for controlling animations etc. Consequently, The Curve Editor documented below can be encountered in many areas of the user interface.

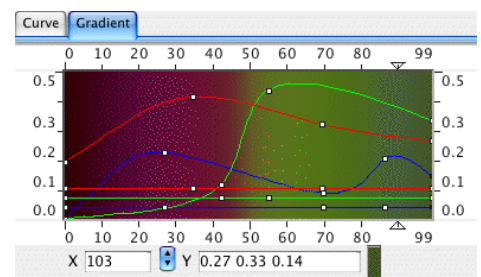
The curve editor can be used in a default state where the 1-4 transformation curves are visible. A curve object consisting of one sub curve is suitable for transforming time, whereas a three-component curve can be used for rgb transformations. The sub curves are colored as 1.=red, 2.=green, 3.=blue, 4.=black.

The Gradient state of the curve editor shows a curve-transformed rgb gradient on the background of the editor. This is especially convenient when designing color gradients for materials or when viewing animated color attributes. It is much more intuitive to see the actual color than three curves representing numeric rgb components.

The curve editor has only a few controls: the actual graphical area showing the curves, and two floating value gadgets for accurate control of selected curve points.



The Curve Editor in the default state



The Curve Editor in the gradient state

Mouse Navigation

The following mouse operations are available for quick navigation:

- *Shift+right mouse button dragging*: Zooms the curve editor in and out. Horizontal dragging zooms the x-scale, vertical dragging the y-scale.
- *Apple+right mouse button dragging*: Pans the curve editor.

Point Editing

Adding points: You can easily add a point to a curve by clicking the curve. A new point is inserted to all visible curves that were hit and which did not have a point at the clicked position.

Deleting points: Select the points and hit Del key or select 'Delete' from the popup menu.

Moving points: Select the points. Then press the left mouse button down on top of a point and move the mouse holding the left button down. Release the left mouse button to finish moving. You can cancel the move operation with the 'Esc' key. It cancels the operation, but it does not undo already moved points.

Point Selection

The point selection rules are similar as in object or 3D handle selection. You can select points by clicking or dragging with the left mouse button. Key modifiers add useful selection functionality. Without a modifier, a selection operation always clears the previous selection. Clicking or dragging empty area deselects all points.

The summary of point selection operations is shown below:

- *LMB click* : Select a new point
- *SHIFT+LMB click* : Add a single point to selection
- *CTRL+LMB click* : Add a selection range from the first previously selected to the clicked point
- *SHIFT+CTRL+LMB click* : Toggle selection
- *ALT+LMB click* : Deselect a point
- *LMB drag* : New selection. Deselects previous points and select new points inside the drag box
- *SHIFT+LMB drag* : Add operation. Adds points inside the drag box to selection
- *SHIFT+CTRL+LMB drag* : XOR operation. Toggles selection state of points inside the drag box
- *CTRL+LMB drag* : And operation. Points inside the drag box AND which were already selected become selected
- *ALT+LMB drag* : Subtract operation. Deselect points inside the drag box

The Popup Menu

Delete Points: Deletes the selected points. Note that a valid curve has always at least two points. The delete tool does not allow point reduction below this minimal amount.

Create Predefined Curve: Initializes an empty curve to a simple default shape. This operation is available for example in animation curve editing where new animated attributes may not have any curve initially defined. The following alternatives are available:

- *Constant Curve:* A horizontal line with a constant y value.
- *45 Degree Curve:* A diagonally ascending line.
- *-45 Degree Curve:* A diagonally descending line.
- *Bell:* A smoothly bending curve with horizontal ends.

Type: Selects the type of the curves from the following alternatives:

- *Nurbs:* A smooth, cubic NURBS curve is fitted to the points. The curve is C2 continuous everywhere, which property is significant in animation use.
- *Polyline:* Linear interpolation between the curve points.
- *PNurbs:* A less continuous curve type than Nurbs. Adapts better to point density variations.
- *Boolean:* A stair curve. Values jump at curve points.
- *Bezier:* The usual cubic Bezier curve, consisting of point quadruples. This curve type allows combination of smoothly curved and linear sections. A Bezier curve is not as continuous as a Nurbs curve at the quadruple joints.

Fit View to Curve: Resets the curve editor after zoom and pan operations so that the current editing range (see 'Set Minimum and Maximum Values' below) matches exactly the window size.

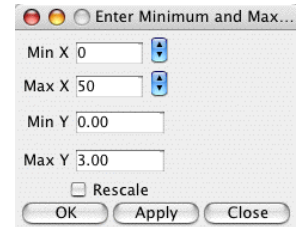
Simplify: Removes unnecessary points of the curves i.e. reduces the point count without changing the curve shape.

Align: Changes the y values of the points between the first and the last selected point of each visible curve so that the points match a straight line.

Set Minimum and Maximum Values: Opens a window for controlling the parameter and output range of the curves. The curve editor allows editing within a certain numeric range. You can define the range using this window.

The scaling dialog has the following options:

- *Min X, Max X, Min Y, Max Y:* The editing range limits. Entering new values does not immediately change the scale; you have to press the 'OK' or the 'Apply' button to set the new limits.
- *Rescale Curve:* If set, the curves are scaled to match the change in editing range.
- *Apply:* Sets the new scale. The scaling window is left open.
- *OK:* Sets the new scale and closes the scaling window.
- *OK:* Closes the scaling window without changing the scale.



Red, Green, Blue, Black Curve: These menus control visibility of the component curves. When editing complex curves, it may be necessary to hide all but one curve. The hidden curves do not interfere with editing actions.

Show All Curves: Makes all component curves visible.

File Rendering Window

The File Rendering Window opens from the File/Render pull down menu (hotkey Apple R). With this window, you can render images and animations to disk files.

The top area of the window contains controls for a file rendering configuration object. The bottom area includes the following gadgets:

- **Time** A progress indicator showing the time used for rendering and the estimated total time.
- **Render Image** This button starts rendering of a single image.



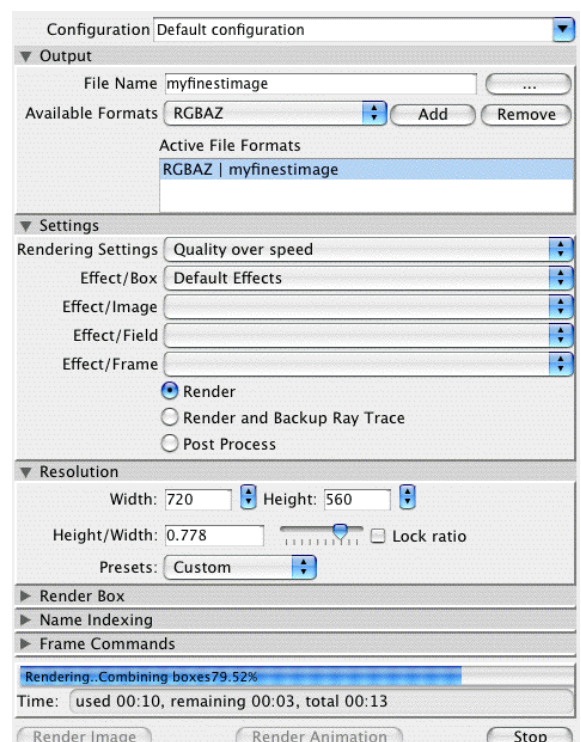
Note

If motion blur or field rendering is active, use the Render Animation button to render all sub images required for one complete frame.

- **Render Animation** Renders all images of the current animation (or a sub interval defined by the Play Range animation option).
- **Stop** Cancels file rendering.

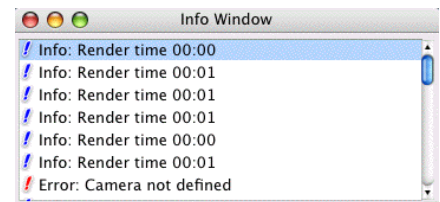
In animation rendering, already rendered separate images will not be deleted. However, an intermediate animation file such as an avi file will be deleted.

In single image rendering, the user will have an option to keep already rendered image boxes, provided that the box rendering feature is used. If the sub boxes are kept, rendering can be continued later by using the existing image sub boxes.



Info Window

The Info Window opens from the Windows pull down menu. The window shows the history of various diagnostic messages given by the program. For example, when encountering an error in scene rendering because of some missing scene element, the warning dialog shows only the latest error message, whereas the message history on the info window may give a better idea of why the rendering failed.

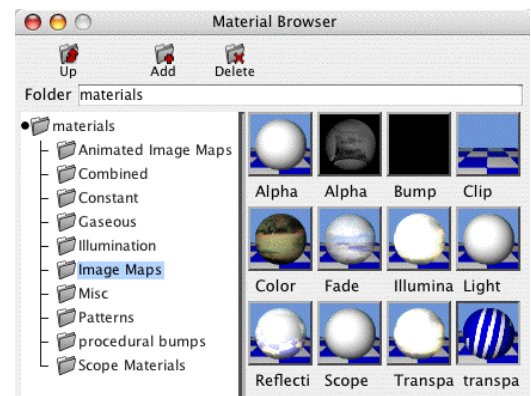


Material Browse Window

The Material Browse Window opens from the material tab pop up menu of the Select Window. With this window, you can load and save materials easily.

The controls of the window are:

- **Up** Opens the parent folder of the current disk folder.
- **Add** Loads the material from the disk file whose icon is selected on the browse window.
- **Delete** Deletes the disk file containing the material whose icon is selected on the browse window.
- **Folder** The name of the current folder.
- **Folder Tree** Displays a disk folder structure. By default, the 'materials' folder is shown.



- **Material Icon Array:** Shows the material files of the selected folder. You can perform the following actions with it:
 - Double clicking a preview icon on the material browse window inserts it in the current library.
 - Drag and dropping a material icon from the browse window into the select window loads the material.
 - Drag and dropping a material icon to a view window loads the material and maps it to the selected geometric objects (just like drag and drop from the select window's material tab to a view window).
 - Drag and dropping a material from the select window to the icon array area saves the material to a disk file using material's name as the file name.

Numeric Window

The Numeric Window is docked to the bottom of the screen in the default configuration. You can open and close it from the View pull-down menu or using the F8 hotkey.

The numeric window is context sensitive. When using various tools of Realsoft 3D, the window makes you available the controls required for performing the operation by exact numeric input instead of mouse clicks. If the numeric window is open, hitting the 'n' (=numeric) key activates the window and moves the input focus to the field which represents the current interaction step. One can then just type the values and hit Enter to proceed to the next input step or to finish the tool. The Tab key moves the focus to the next field without entering a value.

The decimal accuracy and measuring units can be chosen from the File/Preferences pull down menu.

The following controls are always available on the numeric window:

Position	-0.28 0.05 -0.00	Relative	-0.07 0.03 0.00	☐ Polar	Angle 68.96

Numeric window during a rotate operation

Position: The position of the 3D projected mouse pointer in absolute space coordinates. You can always enter a coordinate triplet to this field instead of LMB clicking the mouse. Note that the coordinates are usually projected to the input plane, unless the tool allows free 3D values (for example the NURBS curve tool does this).

Relative: The relative 3D position. When no interaction has been started, the relative position is measured from the grid origin. The grid origin acts therefore as a handy secondary measuring origin, which can be moved easily to a suitable place. When a tool interaction is already going on, the relative coordinates are measured from the previous entered point.

Polar: If set, relative coordinates are measured as angle coordinates (heading, pitching, distance). For example, when drawing a poly line, you can define a 20 mm long, 35 degrees descending line segment using this feature.

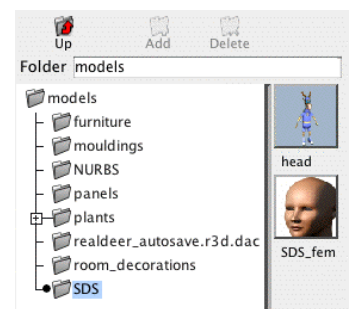
Other context sensitive numeric controls are described in the tool documentation.

Object Browse Window

The Object Browse Window opens from the object tab pop up menu of the Select Window. With this window, you can load and save objects easily.

The controls of the window are:

- **Up** Opens the parent folder of the current disk folder.
- **Add** Loads the object from the disk file whose icon is selected on the browse window.
- **Delete** Deletes the disk file containing the object whose icon is selected on the browse window.
- **Folder** The name of the current folder.
- **Folder Tree** Displays a disk folder structure. By default, the 'models' folder is shown.
- **Object Icon Array:** Shows the object files of the selected folder. You can perform the following actions with it:
 - Double clicking a preview icon on the object browse window inserts it in the current library.
 - Drag and dropping an object icon from the browse window into the select window loads the object.
 - Drag and dropping an object icon to a view window loads the object and translates it to match the current mouse position.
 - Drag and dropping a object from the select window to the icon array area saves the object to a disk file using object's name as the file name.



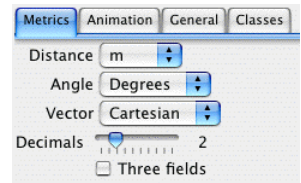
Options Window

The Options Window opens from the File/Preferences pull down menu. The window consists of four tabs described below. The window includes a large set of seldomly changing controls (user specific options, not project specific).

Metrics

Measuring unit and decimal accuracy settings.

- *Distance*: Defines the length measuring unit. All program controls that measure distance or 3D space coordinates use the selected unit.
- *Area*: Defines the area measuring unit.
- *Volume*: Defines the volume measuring unit.
- *Angle*: Selects if angles should be measured using degrees or radians.
- *Vector*: Selects if 3D points should be represented in the usual (x,y,z) Cartesian form or in the Polar coordinate form (heading, pitching, radius). In the latter form, heading defines the projected direction in the horizon plane and pitching defines the angle with the horizon plane. Radius is the distance from the origin.
- *Space*: Defines the world space orientation, Y=up axis or Z=up axis.



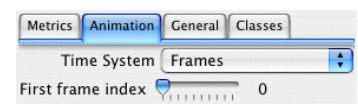
Note

Y = up axis is the native system. All data is internally stored this way. Scripting languages and other low level systems always access the data in the native form.

- *Decimals*: The number of displayed decimal units for all floating point gadgets in the program. For example, a value 1.238 is displayed as 1.24 if decimals is set to two. Note that this does not round the actual value: you can enter floating point values with as many decimals as you like, and they are all taken into account. Note also that the amount of decimals is independent from the measuring unit. Two decimals may be sufficient if you are using millimeters, but 4-5 is more appropriate when using meters.
- *Three fields*: If set, 3D coordinate gadgets consist of three separate sub fields. Then each sub coordinate can be adjusted separately with the dial control and the mouse. Also, changing one coordinate only (leaving two other intact) for a set of selected objects is easier.

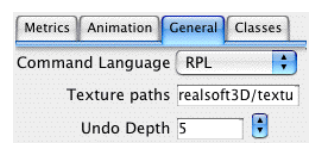
Animation

- *Time system*: You can choose one of the following three alternatives for measuring time:
 - *Frames*: Time is measured using a frame index. The number of frames per second can be selected from Animation Settings/FPS control
 - *Realtime*: Time is measured with three values: minutes, seconds, frames.
 - *Seconds*: Time is measured in seconds.
- *First frame index*: Selects if the first frame index should be zero or one. For example, in a one hundred frame animation, frame indices run from 0 to 99 (indexing starting from zero) or from 1 to 100 (indexing from one).



General

- *Command Language*: Defines which language will be used for scripting (RPL, TCL, JavaScript or another installed language).
- *Texture paths*: You can list your image search paths here. For example, after importing a complex project and placing its textures into a project specific folder, you can make all materials which use textures to work simply by adding the folder into this setting. Several folders can be defined by separating them with a semicolon. For example: 'textures;c:\tilepatterns;mytextures;'

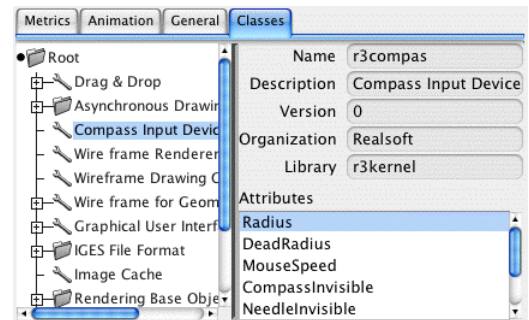


- *Undo Depth*: Defines the amount of successive steps recorder into the undo buffer.

Classes

The Classes tab of the Options window shows all public attributes of the installed program modules. For example, to change the highlight color of point edit handles from the default yellow to green:

- Go to the 'Classes' tab of the options window and find the item 'Wireframe Drawing Context' from the class tree. Activate it by clicking it with the left mouse button.
- The right side of the options window now shows the list of attributes for the wireframe drawing context object. Click 'Handle Fill Color'.
- Below the list, a gadget shows the current value for the selected attribute. Change yellow (1,0,0) to green (0,1,0).



Now close the options window and create an object. Hit the spacebar to activate the point edit mode; then select some points to see that selection is indicated with the new green color.



Note

Many class attributes are read only at the program startup. If you do not see any change after setting a class attribute value, just restart the program.

Property Window

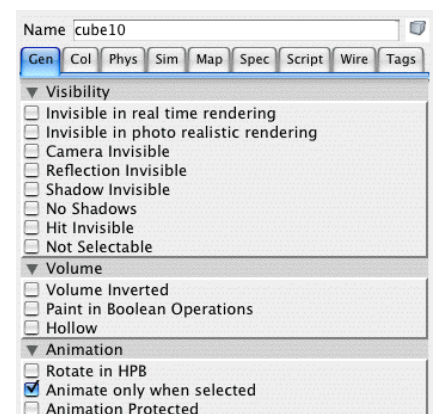
The general purpose Property Window can be opened in many ways:

- From the popup menu 'Properties...' of the select window
- From the popup menu 'Object Properties...' of the view window
- From the 'p' hotkey
- From the 'Windows/Property Window' pulldown menu

In the default user interface, the property window is placed on a floating window. The general purpose property window is context sensitive; it adapts to the type of the selected objects. The select window's visible tab and the objects being selected in the tab together define what controls the property window includes.

There are also some 'sticky' versions of the property window available: they always show properties of selected objects for a certain object layer (independent of select window's tab selection). For example, the following windows which are all available in the 'Windows' pull down menu are such sticky property windows:

- Geometric Properties
- View Properties
- UV View Properties
- VSL Property Window



The property window has only one gadget that is common to all object classes: the **Name** gadget. You can rename one single object as well as several multi-selected objects using it. Remember to hit Tab or Enter key to confirm renaming before closing the property window.

The object property windows can also show properties of multi selected objects. If the values are different, the value of the first selected object is shown using red conflict color. Changing the value sets the same value to all selected objects (hence clearing the conflict state).

The object specific documentation about the property window can be found from the Models section of the reference manual.

Scripting Window

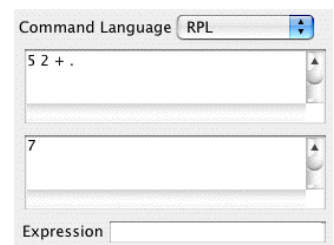
The Scripting Window opens from the 'Windows' pull down menu. Using this window, you can access and change the data of the program environment, loaded projects etc.

Command Language field lists all installed scripting languages. JavaScript is usually available there. The entered commands will be interpreted using the selected language.

You can write script language commands and expressions to the text box under the language selector. The text box provides basic text editing tools. You can naturally paste scripts instead of typing them here.

The second text box shows the output of scripts.

Expression field on the bottom of the window evaluates mathematical expressions.



Select Window

The select windows shows all project and working environment related objects that has been loaded from the hard disk into the computer's memory. You can 'select' or 'activate' items from this window in order to modify them or perform other available operations on them.

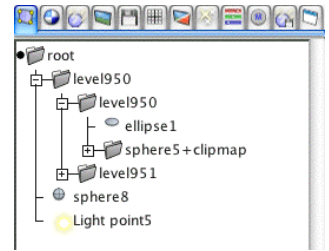
The select window can be hidden and made visible with the F2 hotkey, and from the 'View/SelectWindow' pull down menu.

The top area of the select window contains a list of opened projects. The list can be hidden and made visible using the **Projects...** check box. The current project can be changed by opening the project list and clicking an item. Usually there is at least two projects included: the current working project and a material preview project. The 'File/New' pull down menu adds a new empty project into the list and 'File/Close' closes the selected project (note: material preview cannot be closed). Closing the last working project automatically creates a new empty one. It should be noted that 'File/Load' loads into the current project instead of adding a new project.

A project can be renamed using the **Name** field below the project list. Renaming a project will change the suggested file name when the project is saved next time.

Geometric Objects

The Hierarchy Tree: Displays the tree like structure of the geometric objects. Geometric objects are usually grouped using level objects to obtain a sensible and manageable object hierarchy. A suitable part of the total structure can be activated here, and parts can be moved in hierarchy using drag and drop. See the Mouse Operation documentation below for details.



Filter: With the filter string, you can limit the set of objects displayed on the hierarchy tree using object names and the wildcard character '*'. For example, if you want to only see objects directly below the root level and whose name starts with the letters 'sp' (spheres, for example), set the filter string to '/root/sp*'. The filtering ability helps in searching objects from a complex model.

Mouse Operations

Dragging one or several selected objects with the left mouse button moves the objects in the hierarchy tree. Arrow icons indicate the new location already before releasing the mouse button. The following variations are available:

- *LMB dragging an item:* Moves an item to a new location in hierarchy.
- *Alt+LMB dragging an item:* Creates an instance of the selected object(s) and moves it to a new location in hierarchy.
- *Apple+LMB dragging an item:* Creates a copy of the selected object(s) and moves it to a new location in hierarchy.

Left mouse button dragging without hitting an object at the drag start point performs object selection.

- *LMB dragging:* All objects inside the drag box will become selected. Possible previous selection is lost.
- *Shift+LMB dragging:* All objects inside the drag box will be added to previous selection.
- *Apple+LMB dragging:* Boolean AND operation 'previous selection AND new selection'. All objects inside the drag box that were also previously selected will remain selected. All other objects become deselected.
- *Alt+LMB dragging:* Subtracts all objects inside the drag box from selection.
- *Shift+Apple+LMB dragging:* XOR operation. Inverts selection of objects inside the drag box. Selected objects become unselected and vice versa.



Note

Only objects inside the drag box change their selection state.

Left mouse button clicking activates objects. Clicking an empty area deselects all objects.

- *LMB click:* Selects the clicked object. Other objects become deselected.
- *Shift+LMB click:* Adds the clicked object to the previous selection.
- *Apple+LMB click:* Range selection. Adds all objects between the previous selected object (closest upwards in the hierarchy tree in the same hierarchy level with the clicked object) and the clicked object to the selection. If there are no other selected objects in the same hierarchy level, selection starts from the beginning of the level.
- *Alt+LMB click:* Deselects the clicked object.
- *Shift+Apple+LMB click:* Toggles (=inverts) selection of the clicked object.

Double clicking activates some additional tools:

- *Slow LMB double click*: Activates the 'Rename' tool.
- *Quick LMB double click*: Opens the object property window.
- *Shift + quick LMB double click*: Hides the hierarchy above a clicked level object. The level will appear as the root of the hierarchy. Another Shift double click will restore the original root.

The contents of the select window can be scrolled by dragging with the middle mouse button, or by the right mouse button dragging Apple key held down.

Hotkeys

Most hotkey events at the select window are forwarded to the active view window. For example, hitting the 'O' key renders the active view. Only the 'tab' key has a special use: it sets the current level to the selected level. If the selected object is not a level object, nothing happens. The current level defines the hierarchical destination for created, loaded or pasted objects.

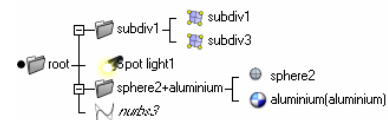
The Popup Menu

The popup menu of the object geometry tab is context sensitive. New menu items appear depending on object selection. For example, 'Rename' menu is not available if all objects are deselected. Another example: 'Open' menu appears only when one or more level objects are selected.

As usual, plugin products may insert additional items to the popup menu of the select window.

- **New** These sub items create some special purpose objects into the object hierarchy.
 - *3D Measure*
 - *Angle Measure*
 - *Measure*
 - *Position Mark*
 - *Level*
 - *Rotator*
 - *SunMoon*
 - *Fractal Tree*
 - *Post Image*
 - *Plant Seed*
 - *Choreography*
 - *Tube*
 - *Interpolator*
 - *Collision Deformer*
 - *Inverse Kinematics*
 - *Drag*
 - *Morphing*
 - *Metaball*
 - *Creator*
 - *Foot Print*
 - *Field Evaluator*
- **Open (Close)** The Open menu opens the selected level object so that its sub objects become visible. Sub objects can then be selected individually. The Close menu does the opposite i.e. hides the sub hierarchy.
- **Open All (Close All)** Opens the sub hierarchy recursively. Sub objects and their sub objects etc. become visible on the select window. Close all closes the sub hierarchy recursively.
- **Make Current** Makes the selected level-type object the current level. The 'current level' concept defines where new created, loaded or pasted objects should be placed in hierarchy.
- **Edit** Puts an object to the editing mode. Usually objects reveal some object specific handles in the editing mode, allowing the user to modify the characteristic geometric details of the object.
- **Drop to a Level** Creates a new level object and inserts all selected objects into it. The new level will be placed to the location in hierarchy where the first selected object was.
- **Set Filter** Copies the full hierarchy path from the selected object to the name filter string. The string can then be modified as appropriate. The filter tool allows the user to display only a suitable subset of the hierarchy using string pattern matching. This is a useful option when searching items from a complex hierarchy.
- **Cut** Cuts the selected objects from the object hierarchy to the Realsoft 3D geometry clipboard.

- **Copy** Copies the selected objects from the object hierarchy to the Realsoft 3D geometry clipboard. The selected objects stay in hierarchy.
- **Paste** Pastes geometric objects from the Realsoft 3D clipboard to the current level.
- **Duplicate** Creates duplicates of the selected objects. Each duplicate is placed to the same hierarchy level with its original object.
- **Delete** Deletes the selected objects. The clipboard will not change.
- **Copy to a File** Opens a file browser. Selected objects are then saved to a defined disk file.
- **Paste from a File** Opens a file browser. Objects from a selected disk file will be loaded to the current level.
- **Select Method**
 - *Don't select new objects* Created/loaded/pasted objects will not become automatically selected.
 - *Select newly created* Created/loaded/pasted objects will become automatically selected. Previous object selection is lost.
 - *Add to selection* Created/loaded/pasted objects will become automatically selected. Previously selected objects remain also selected.
- **Balanced Tree** Displays the hierarchy as a horizontally organized tree.
- **Rename** Activates a rename box for the selected object.
- **Properties..** Opens the object property window.



Balanced hierarchy mode

- **Make Choreographable** Makes an object animated. The time line and other animation properties can then be edited with the Choreography Window.



Note

Modifying an object property in the animation recording mode makes the object also choreographable.

- **Load Choreography** Loads animation properties from a disk file and attaches the loaded information to selected objects (only common attributes become utilized - a sphere radius animation will not attach to a cube).
- **Add New Choreography** Adds a new empty choreography to the selected objects. The new choreography will be the active one i.e. animation recording actions will be stored to it. Multiple choreographs can be combined in a weighted manner.
- **Delete Choreographs** Removes all animation information from the selected objects.

Drag and Drop

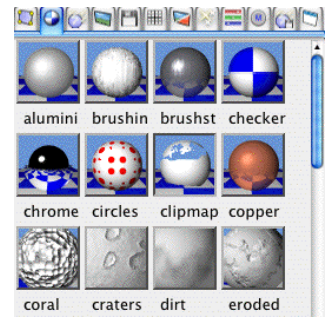
Dragging an object from the geometry tab of the select window to a view window performs an auto focus operation: the view is panned, rotated and zoomed so that the object fills most of the view window. The camera object is an exception: the view window fetches the camera view of the dropped object.

See also: Reference Manual/Models/Geometric Objects.

Materials

The second tab of the select window contains the current material library. Materials are objects which control object shading. The *Current material Library* means the set of materials loaded into the computer memory for use in the project under construction.

The program installation contains a set of predefined materials stored into the 'materials' folder and its sub folders. Part of these materials are loaded with the default startup file. When building a project, the user needs only a certain subset of materials, which are also often customized for the project needs. The current material library is a suitable place to store such project specific materials. When the users saves the project, the current material library is saved as a part of the project data, and the original material templates remain intact in the 'materials' folder.



A typical material usage workflow goes as:

- Start the program and model a scene.
- See if the default startup loaded library contains suitable materials and assign them to objects.
- Apply the 'Purge' feature to remove unused materials. This saves disk space when later saving the project.
- Load (or build) the rest of the required materials from disk files of the 'material's folder.
- Save the project

The Popup Menu

- **New** Tools for inserting a new material into the current library.
 - *From Template* Opens a file browser for loading a material that has been saved to a disk file.
 - *VSL Material, VSL Procedure Library, Level Material, V3 Material* Creates new empty materials to the current library.
- **Browse Material Library** Opens a special preview icon based window for loading disk based materials. Unlike the 'New/From Template' menu, this method allows you to load several materials at the same time. With this window, you can load a material from the disk file in one of the three following ways:
 - Double clicking a preview icon on the material browse window inserts it in the current library.
 - You can also drag and drop a material from the browse window into the select window.
 - Drag and dropping a material from the browse window to a view window loads the material and maps it to the selected geometric objects (just like drag and drop from the select window's material tab to a view window). Drag and dropping a material from the select window to the browse window saves the material to a disk file using material's name as the file name.
- **Cut** Cuts the selected material from the material library to the Realsoft 3D material clipboard.
- **Copy** Copies the selected material to the Realsoft 3D material clipboard. The selected material stays in the library.
- **Paste** Pastes the material from the Realsoft 3D clipboard to the current library.
- **Duplicate** Creates a duplicate of the selected material. The duplicate is placed to the end of the material library.
- **Delete** Deletes the selected material. The clipboard will not change.
- **Delete All** Deletes all materials. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every material has a unique name.

- **Purge** Deletes all materials that have not been assigned to objects of the scene.
- **Copy to a File** Opens a file browser. The selected material is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The materials from a selected disk file will be loaded to the current library (this is the same as New/From Template - the menu is here for consistency).
- **Sort Alphabetically** Rearranges the materials into alphabetical order.
- **Map** The sub items of this menu create material mapping objects. A mapping object attaches the selected material to selected geometric objects using a suitable geometric 'wrapping' transformation. After the tool activation, most mapping types require some mouse clicks which define the mapping geometry. See also Mapping Objects documentation.
- **Rename** Activates a rename box for the selected material.
- **Show Previews** If set, material preview icons are shown on the select window. If cleared, only the names are shown.
- **Properties..** Opens the object property window.
- **Make Choreographable** Makes an material object animated. The time line and other animation properties can then be edited with the Choreography Window.



Note

Modifying a material property in the animation recording mode makes the material also choreographable.

- **Load Choreography** Loads animation properties from a disk file and attaches the loaded information to the selected material.
- **Add New Choreography** Adds a new empty choreography to the selected material. The new choreography will be the active one i.e. animation recording actions will be stored to it. Multiple choreographs can be combined in a weighted manner.
- **Delete Choreographs** Removes all animation information from the selected material.

Drag and Drop

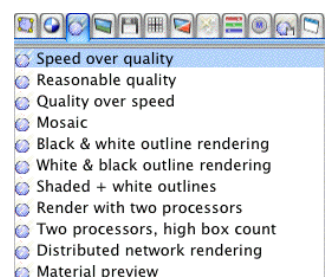
Dragging a material from the select window to a view window auto maps the material to the selected geometric objects. The recommended mapping type is used for each selected object, as specified in the material properties. If no objects are selected, a mapping is inserted to the current geometry level. If a level object is selected, the first sub object is used for mapping recommendation calculations.

See also: Reference Manual/Models/Material Objects.

Render Settings

A render settings object includes options for photorealistic rendering. The render settings library is common to all projects: if you have multiple projects open, they all share the same collection.

The Popup Menu



- **New** Creates a new render settings object, which can be customized as appropriate.
- **Cut** Cuts the selected settings object to the Realsoft 3D render settings clipboard.
- **Copy** Copies the selected settings object to the Realsoft 3D render settings clipboard. The selected object stays in the library.
- **Paste** Pastes the settings object from the Realsoft 3D clipboard to the render settings library.
- **Duplicate** Creates a duplicate of the selected settings object. The duplicate is placed to the end of the library.
- **Delete** Deletes the selected object. The clipboard will not change.
- **Delete All** Deletes all objects. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every setting has an unique name.
- **Purge** Deletes all unused setting objects (not used by view windows or file render objects).
- **Copy to a File** Opens a file browser. The selected render settings object is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The object from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected render settings object.
- **Properties..** Opens the object property window.

Drag and Drop

Dragging a render settings object from the select window to a view window selects the render settings for the view.

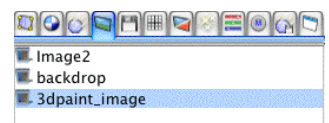
See also: Reference Manual/Models/Rendering Settings.

Images

The image library is projects specific: if you have multiple projects open, they all have their own set of images.

The Popup Menu

- **New** Creates a new empty image object.
- **Cut** Cuts the selected image to the Realsoft 3D image clipboard.
- **Copy** Copies the selected image object to the Realsoft 3D image clipboard. The selected image stays in the library.
- **Paste** Pastes an image object from the Realsoft 3D clipboard to the current image library.
- **Duplicate** Creates a duplicate of the selected image object. The duplicate is placed to the end of the image library.
- **Delete** Deletes the selected image object. The clipboard will not change.
- **Delete All** Deletes all objects. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every image has an unique name.



- **Purge** Deletes all unused image objects (which are not used, for example, as view backdrops).
- **Copy to a File** Opens a file browser. The selected image object is then saved to a defined disk file.



Note

This does not save the actual RGB data to the defined file, only the 'wrapper' object!

- **Paste from a File** Opens a file browser. The object from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected image object.
- **Properties..** Opens the object property window.
- **Make Choreographable** Makes an image object animated. The time line and other animation properties can then be edited with the Choreography Window.



Note

Modifying a property in the animation recording mode makes the object also choreographable.

- **Load Choreography** Loads animation properties from a disk file and attaches the loaded information to selected image.
- **Add New Choreography** Adds a new empty choreography to the selected object. The new choreography will be the active one i.e. animation recording actions will be stored to it. Multiple choreographs can be combined in a weighted manner.
- **Delete Choreographs** Removes all animation information from the selected image.

Drag and Drop

Dragging an image object from the select window to a view window sets the view backdrop image.

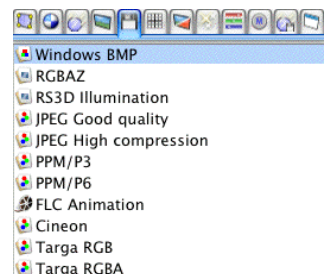
See also: Reference Manual/Models/Image Objects.

Output Objects

An output object defines a file format or another output target with customizable channel bindings. The output object library is a shared one: if you have multiple projects open, they use the same library.

The Popup Menu

- **New** Creates a new output object. The sub menu items include the installed output object types (supported image file formats).
- **Cut** Cuts the selected output object to the Realsoft 3D output object clipboard.
- **Copy** Copies the selected output object to the Realsoft 3D output object clipboard. The selected object stays in the library.
- **Paste** Pastes the output object from the Realsoft 3D clipboard to the current library.



- **Duplicate** Creates a duplicate of the selected output object. The duplicate is placed to the end of the output object library.
- **Delete** Deletes the selected output object. The clipboard will not change.
- **Delete All** Deletes all objects. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every output object has a unique name.
- **Purge** Deletes all output objects, which are not used by file render configurations etc.
- **Copy to a File** Opens a file browser. The selected output object is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The output objects from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected output object.
- **Properties..** Opens the object property window.

Drag and Drop

You can add output objects for file rendering by dragging and dropping them from the select window to the active output list of the file rendering window.

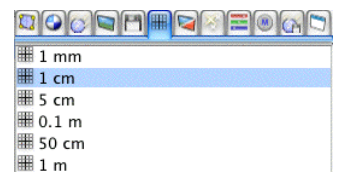
See also: Reference Manual/Models/Output Objects.

Grids

Grid objects round convert mouse coordinates to a defined base interval division. The grid library is project specific: if you have multiple projects open, each has its own set of grids.

The Popup Menu

- **New** Creates a new grid object to the grid library.
- **Cut** Cuts the selected grid to the Realsoft 3D grid clipboard.
- **Copy** Copies the selected grid to the Realsoft 3D grid clipboard. The selected grid stays in the library.
- **Paste** Pastes the grid object from the Realsoft 3D clipboard to the current grid library.
- **Duplicate** Creates a duplicate of the selected grid. The duplicate is placed to the end of the library.
- **Delete** Deletes the selected grid. The clipboard will not change.
- **Delete All** Deletes all objects. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every grid has a unique name.
- **Purge** Deletes all unused grids.
- **Copy to a File** Opens a file browser. The selected grid is then saved to a defined disk file.



- **Paste from a File** Opens a file browser. The grid object from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected grid object.
- **Properties..** Opens the object property window.

Drag And Drop

Dragging a grid object to a view window selects the current grid for the view and activates grid drawing and snapping.

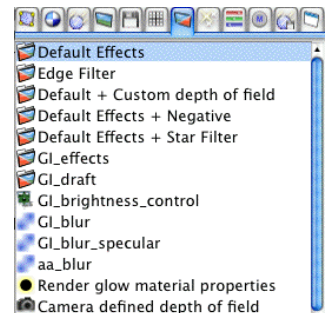
See also: Reference Manual/Models/Grid Objects.

Post Image Effects

Post image effects manipulate the pixel information from the ray tracer - they are 'raster' effects. The post image library is project specific: if you have multiple projects open, each uses its own private effects.

The Popup Menu

- **New** Creates a new post image effect. Sub menu items show the all installed image effects.
- **Cut** Cuts the selected image effect to the Realsoft 3D image effect clipboard.



- **Copy** Copies the selected image effect to the Realsoft 3D image effect clipboard. The selected effect stays in the library.
- **Paste** Pastes the image effect from the Realsoft 3D clipboard to the current effect library.
- **Duplicate** Creates a duplicate of the selected image effect. The duplicate is placed to the end of the library.
- **Delete** Deletes the selected effect. The clipboard will not change.
- **Delete All** Deletes all effects. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every effect has an unique name.
- **Purge** Deletes all unused image effects.
- **Copy to a File** Opens a file browser. The selected image effect is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The image effect objects from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected image effect.
- **Properties..** Opens the object property window.
- **Make Choreographable** Makes an object animated. The time line and other animation properties can then be edited with the Choreography Window.



Note

Modifying a property in the animation recording mode makes the object also choreographable.

- **Load Choreography** Loads animation properties from a disk file and attaches the loaded information to selected image effect.
- **Add New Choreography** Adds a new empty choreography to the selected object. The new choreography will be the active one i.e. animation recording actions will be stored to it. Multiple choreographs can be combined in a weighted manner.
- **Delete Choreographs** Removes all animation information from the selected image effect.

Drag And Drop

Dragging a post image object to a view window selects the post effect configuration per render box for the view. Dragging post effects to a view inserts them into the post effect configuration used by the view per render box.

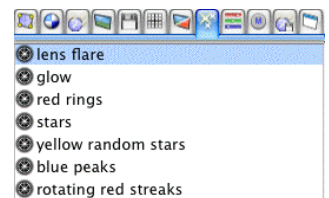
See also: Reference Manual/Models/Post Image Effects.

Post Particle Effects

Post particle effects define how particles is rendered in post processing phase. In other words, you can assign different appearances to particles using post particle effects. The post particle effect library is project specific: if you have multiple projects open, each project has its own set of effects.

The Popup Menu

- **New** Creates a new particle effect. Sub menu items show the list of installed particle effects.
- **Cut** Cuts the selected particle effect to the Realsoft 3D particle effect clipboard.
- **Copy** Copies the selected particle effect to the Realsoft 3D particle effect clipboard. The selected particle effect stays in the library.
- **Paste** Pastes a particle effect from the Realsoft 3D clipboard to the current library.
- **Duplicate** Creates a duplicate of the selected particle effect. The duplicate is placed to the end of the library.
- **Delete** Deletes the selected particle effect. The clipboard will not change.
- **Delete All** Deletes all particle effects. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every particle effect has an unique name.
- **Purge** Deletes all unused particle effects.
- **Copy to a File** Opens a file browser. The selected particle effect is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The particle effect from a selected disk file will be loaded to the current library.



- **Rename** Activates a rename box for the selected particle effect.
- **Properties..** Opens the object property window.
- **Make Choreographable** Makes a particle effect animated. The time line and other animation properties can then be edited with the Choreography Window.



Note

Modifying a particle effect property in the animation recording mode makes the particle effect also choreographable.

- **Load Choreography** Loads animation properties from a disk file and attaches the loaded information to selected particle effect.
- **Add New Choreography** Adds a new empty choreography to the selected particle effect. The new choreography will be the active one i.e. animation recording actions will be stored to it. Multiple choreographs can be combined in a weighted manner.
- **Delete Choreographs** Removes all animation information from the selected particle effect.

Drag And Drop

Dragging a post particle effect to a view window auto maps the effect to the selected geometric objects. This normally makes sense only when the selected objects are particles.

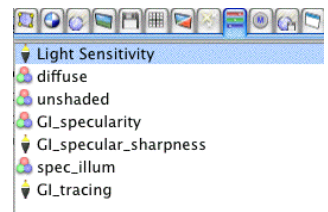
See also: Reference Manual/Models/Particle Effects.

Channels

Channels are user definable rendering attributes. The channel library is shared between all projects. For example, the material preview project automatically finds all new channels of a custom material shader.

The Popup Menu

- **New** This menu has three sub items for creating new float value, color and vector channels.
- **Cut** Cuts the selected channel to the Realsoft 3D channel clipboard.
- **Copy** Copies the selected channel to the Realsoft 3D channel clipboard. The selected channel stays in the library.
- **Paste** Pastes a channel from the Realsoft 3D clipboard to the current library.
- **Duplicate** Creates a duplicate of the selected channel. The duplicate is placed to the end of the channel library.
- **Delete** Deletes the selected channel. The clipboard will not change.
- **Delete All** Deletes all channels. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every channel has an unique name.
- **Purge** Deletes all unused channels.
- **Copy to a File** Opens a file browser. The selected channel is then saved to a defined disk file.



- **Paste from a File** Opens a file browser. The channels from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected channel.
- **Properties..** Opens the object property window.

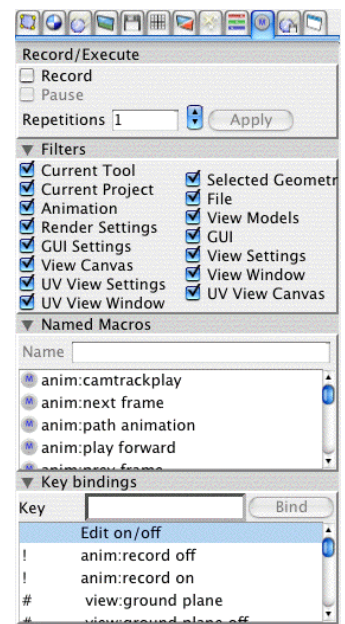
See also: Reference Manual/Models/Channels.

Macros

Macros store user defined action sequences. Recording several frequently needed steps into a macro often improves working efficiency. Macros are shared between projects: if you load a project which includes a macro section, all opened projects use the new macro set.

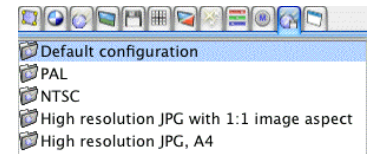
The Controls of the Macro Tab

- **Record** Turns macro recording on/off.
- **Pause** Turns macro recording temporarily off. You can then perform operations that do not record into the macro and continue recording again later.
- **Repetitions** Defines how many times the current macro is executed when the Apply button is pressed.
- **Apply** Executes the recorded macro ('Repetitions' times).
- **Filters** Filter options define what kind of actions get recorded into macro. For example, if 'View Canvas' option is disabled, activating the OpenGL fog option does not record into the macro.
- **Name** You can rename the recorded macro with this gadget. If you want to store a recorded macro with your working environment, you have to rename it.
- **Macro List** The list of all named macros. The macro collection is loaded from the 'Macros' section of a project file. You can add a new macro to the list by recording one and renaming it appropriately. You can execute a macro by dragging it to a view window.
- **Key** A field for defining a hotkey combination that executes the selected macro:
 - Record a macro and rename it to a unique name, or select an existing named macro from the list.
 - Select the geometric object which should activate a context sensitive macro. If the macro is not object specific, deselect all geometric objects.
 - Activate the 'Key' field and press the key combination you want to use for the macro.
 - Make sure that the appropriate macro is selected on the macro list above and press the 'Bind' button.
 - To use the new key binding at a later date, save the project with 'Macros' section.
- **Bind** Creates a new hot key binding between the 'Key' field contents and the currently selected named macro.
- **Key Binding List** The list of all macro key bindings. Use the popup menu to delete undesired bindings.



File Rendering Configurations

A file rendering configuration combines render settings, post processing configurations, output resolution and output file formats together into a ready-to-use package. For example, you can define a PAL video rendering configuration. The file rendering configuration library is shared between all projects: multiple opened projects use the same collection.



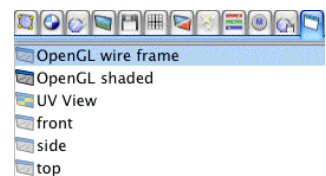
The Popup Menu

- **New** Creates a new rendering configuration with some default settings.
- **Cut** Cuts the selected rendering configuration to the Realsoft 3D file rendering clipboard.
- **Copy** Copies the selected rendering configuration to the Realsoft 3D clipboard. The selected rendering configuration stays in the library.
- **Paste** Pastes a rendering configuration from the Realsoft 3D clipboard to the current library.
- **Duplicate** Creates a duplicate of the selected rendering configuration. The duplicate is placed to the end of the library.
- **Delete** Deletes the selected rendering configuration. The clipboard will not change.
- **Delete All** Deletes all configurations. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every configuration has an unique name.
- **Purge** Deletes all unused configuration objects.
- **Copy to a File** Opens a file browser. The selected rendering configuration is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The rendering configurations from a selected disk file will be loaded to the current library.
- **Rename** Activates a rename box for the selected rendering configuration.
- **Properties..** Opens the object property window.

See also: Reference Manual/Models/File Rendering Configurations.

View Models

A view model stores settings for a view window. Therefore, just like the GUI, the view model collection is shared between opened projects. When you swap from a project to another one, the GUI and the view model library it uses remain the same. When you load a new project with a customized GUI, all opened projects will use the new GUI and new view models.



The Popup Menu

- **New** Creates a new view model object.
- **Cut** Cuts the selected view model to the Realsoft 3D view model clipboard.
- **Copy** Copies the selected view model to the Realsoft 3D view model clipboard. The selected view model stays in the library.
- **Paste** Pastes a view model from the Realsoft 3D clipboard to the current library.

- **Duplicate** Creates a duplicate of the selected view model. The duplicate is placed to the end of the library.
- **Delete** Deletes the selected view model. The clipboard will not change.
- **Delete All** Deletes all view models. The program displays a warning before deletion.
- **Delete Duplicates** Removes objects which duplicate a name of an earlier object. After the operation, every view model has an unique name.
- **Purge** Deletes all unused view models.
- **Copy to a File** Opens a file browser. The selected view model is then saved to a defined disk file.
- **Paste from a File** Opens a file browser. The view models from a selected disk file will be loaded to the current view model library.
- **Rename** Activates a rename box for the selected view model.
- **Properties..** Opens the object property window.

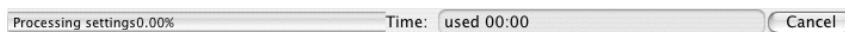
Drag And Drop

Dragging a view model to a view window assigns the view model for the view. If the view model is already in use, a duplicate is created first (view models cannot be shared).

See also: Reference Manual/Models/View Models.

Status Window

The Status Window is available as a docked horizontal bar at the bottom of the default user interface. The window can be hidden/made visible using the View pull down menu, or with the hotkey 'F9'.



The main purpose of the status window is to give various instructions to the user. When the user starts a tool, the window instructs how to perform the basic steps of the tool. However, because of space limitations, such information is shown in a compact form.

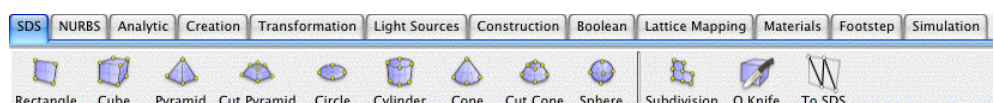
The window also shows a progress indicator during time consuming operations. For example, when ray trace rendering a view window, the progress indicator starts showing rendering time statistics. The Cancel button is also enabled and pressing it cancels the rendering.

Tool Bar

The Tool Bar Window is docked to the top of the default user interface. It contains a set of tabs, providing access to most of the tools available in the program. The contents of the toolbar is user configurable.

Toolbar, just like other program windows, can be scrolled by dragging it with the right mouse button 'Apple' key held down. Alternatively, the middle mouse button can be used for dragging without any key modifiers. The tool bar can be hidden and made visible from the pull down menu View/Tabbed tool bar or with the hotkey 'F3'.

Usually the toolbar works in conjunction with the tool control bar. When a tool is activated, the tool control bar shows the tool specific options.



Tool Control Bar

The Tool Control Bar is docked below the toolbar. It is context sensitive: when a tool is activated, the tool control bar shows the tool specific options. Or, when an object is selected, the tool control bar shows tools which are specific to that particular object type.

If the contents of the control bar do not fit inside the visible area, you can scroll it by dragging it with the right mouse button 'Apple' key held down, or with the middle mouse button.

The tool control bar can be hidden and made visible from the pull down menu View/Control bar or with the hotkey 'F4'.



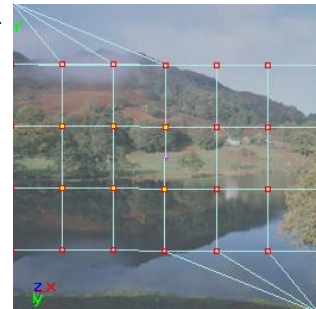
The tool control bar showing the context of the NURBS sphere tool

UV Editor

The UV editing window opens from the 'Windows/UV editor' pull down menu. F11 is the hotkey for opening the window.

With this window, you can edit:

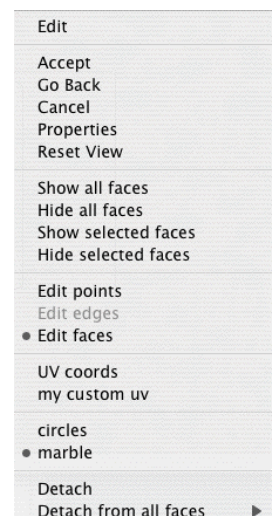
- UV coordinates of NURBS meshes
- UV coordinates and other vertex variables of SDS objects
- UV coordinates of UV Set objects
- Face UV's of SDS objects
- UV coordinates of trim curves



Editing UV coordinates of a SDS sphere

The popup menu of the UV editor is context sensitive. Below is a picture of the menu when editing a subdivision surface which has two vertex UV channels and two face mapped materials. The purpose of these menus are described below.

- **Edit:** Turns edit mode on/off. Edit mode must be enabled before UV editing can take place.
- **Accept, Go Back, Cancel:** The standard menus for interaction control.
- **Properties:** Opens the UV editor property window where various editor settings can be changed.
- **Reset View:** Resets the windows scale. The window will show exactly the unit square of the UV plane.
- **Show all faces:** Makes all faces being edited visible. This includes all SDS faces if you are editing vertex UV's. In face UV editing, all faces that have the currently selected material mapped on them become visible. With this and the three following menus, you can reduce excessive UV mesh complexity so that a suitable subset of data is left visible for editing.



- **Hide all faces:** Makes all faces being edited invisible. After applying this menu, you can make a suitable subset of faces visible with the 'Show selected faces' menu.



Note

You can perform the selection in a normal 3D view. Hiding faces in the UV editor does not make them invisible in a 3D view.

- **Show selected faces:** Makes all faces that are currently selected visible.
- **Hide selected faces:** Makes all faces that are currently selected invisible.
- **Edit Points:** Switches the UV editor to edit point handles.
- **Edit Edges:** Selects editing by using edge handles.
- **Edit Faces:** Editing with face handles.
- **UV coords and my_custom_uv:** This menu group shows all UV type vertex variables which are attached to the object. You can change the editing target to one of the vertex variables using these menu items.
- **circles and marble:** This menu group lists all materials that are face mapped to the selected SDS object. You can select from the menu which face material UV's to edit.
- **Detach:** Removes the material indicated by the menu group above (marble in this example) from the selected SDS faces.
- **Detach All:** This menu lists all attached face materials. You can remove the desired material attachment from all faces (regardless of the face selection).

Navigation

The following navigation operations are available:

- Apple + RMB dragging (or middle mouse button dragging) pans the UV editor
- Shift + RMB dragging zooms the UV editor in and out

UV Coordinate Types

NURBS surfaces and trim curves have only one UV channel available. Actually a NURBS surface may not have any UV channel at all, but you can attach and initialize the channel using the pull down menu 'Methods/NURBS mesh/Define UV'. When editing this kind of objects, there is no need to select what to edit in the UV editor.

The situation is more complex when editing a SDS object. This object type not only allows several UV channels mapped to vertices, but also supports face mapped materials which have their own parametrization. Vertex UV channels store one UV value per vertex so that adjacent faces share the same UVs. Shared values make this parametrization type automatically continuous over face boundaries.

A face mapped material stores a private UV value for each vertex per face per material. The advantage is that faces can be torn apart in the UV space. A typical disadvantage is that face mapped materials do not run continuously over face boundaries.

UV Editor Backdrop

- The object has a direct UV texture (defined in Property window/Col tab/Texture field). The backdrop is drawn using a maximal accuracy, limitation being the resolution of the UV editor or the texture image itself.

- When editing of a face mapped material of a SDS object is started, the program automatically computes a backdrop image. The backdrop accuracy is defined by the Property window/Wire tab/Texture Quality field.
- You can also drag and drop any image from the select window's image tab onto the UV editor.



Note

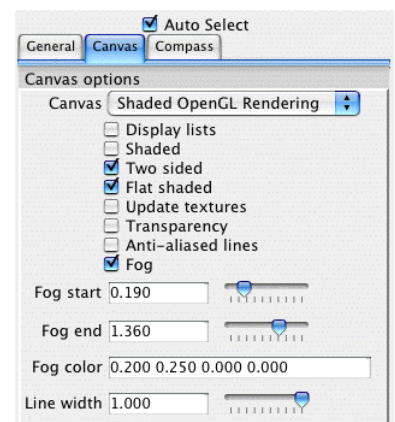
Make sure that you don't have a SDS object selected and in edit mode, because you can assign images to selected faces the same way, too. Alternatively, the backdrop image can be created on the select window and activated using a UV view property window.

The contrast of the backdrop can be controlled from the UV view property window. Usually it is necessary to fade the backdrop suitably so that the UV mesh wireframe and handles become better visible over the background.

UV Editor Properties

You can edit the settings of the UV editing window by selecting its UV view model from the select window and by opening the standard property window. The property window can also be opened from the popup menu of the uv view.

The controls of the UV editor property window are a subset of the controls of the View property window.



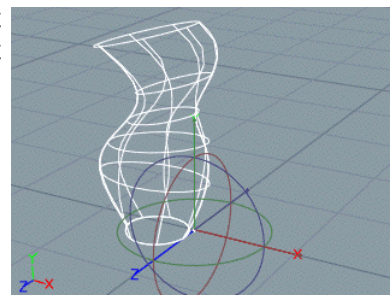
View

The view window is probably the most important element of the user interface. It shows you the geometric structure of the scene and projects mouse coordinates into the 3d space. The default user interface contains one large, docked view window, but you can open as many view windows as you like. An additional floating view window opens from the 'Windows/View Window' pull down menu; alternative user interfaces such as the quadview contain multiple view windows which show the scene simultaneously from several directions.

The popup menu of the view window is partly context sensitive. The topmost menu items are related to the program tools and therefore the menu set depends on the current stage of the possibly ongoing tool interaction.

The menu structure is:

- *Accept*: Finishes the current interaction
- *Go Back*: Undoes the last step of current interaction
- *Cancel*: Cancels the current interaction
- *Again*: Restarts previous interaction
- *Render*:
 - *Wire*: Refreshes the view window using the selected real time drawing system



- *Ray Trace*: Activates photorealistic ray trace rendering into the view window
- *Backup Ray Tracing*: Enables/disables a backup buffer for ray trace rendering. The result of rendering is stored into a disk file, which includes all channels needed by the selected post effects. The deep image can be viewed later and post effects can be repeatedly computed using it as the source data.
- *Post Process*: Renders only the post effects. If a backup buffer is available, the post processing system uses it instead of an image coming from the ray tracer.
- *Post Process File*: Applies the view's post effect configuration to an image file on disk. Result is shown on the view window. The image file is not scaled: if the source image does not fit the view window, additional parts become clipped.
- *Define Render Box*: Activates render box interactor. The user can LMB drag a rectangle whose contents will be ray traced. All later ray trace renderings will be limited to the defined area until the render box is deactivated.
- *Box Active*: Activates/deactivate the render box
- *Save to File*: Saves the contents of the current render box (or the whole view if render box is unused) to a given disk file.
- *Animation Preview*: Plays the animation and saves view window contents to a given image/animation file. after each frame. The format can be selected from the view property window (.avi by default). A prerendered avi file can then be played at constant speed regardless of the scene complexity. You can stop the preview rendering using the play button of the animation window.
- *Cancel Rendering*: Stops the current rendering operation.
- *Color, ...*: The channels of the backup buffer are listed at the end of the Render menu. Picking a channel item from the menu displays the channel contents on the view.
- *Handles*: The top level switches for the general object handles.
 - *Bounding Box*: Enables drawing of a bounding box handle around the selected object. You can scale the selected objects by LMB-dragging the corners of the bounding box and move them by dragging the bounding box edges. The bounding box handle works also as a useful selection indicator in shaded OpenGL, because it is drawn outside (possibly obscuring) surfaces.
 - *Rotate*: Enables drawing of the object space rotate, translate and scale handles. These handles are drawn according to the selected object. If multiple objects are selected, the handles are drawn for the first object only. The three circular handles rotate the object. The three coordinate axis lines move the object in one dimension at a time. The end knobs of axis lines scale the object in the respective directions. The origin point of the handle system is the pivot point translation handle. All these handles can be used either by LMB-dragging them, which is a quick but inaccurate way. The handles can also be activated by a LMB click and then used with 2 clicks more. This method is more accurate because drag-snapping can be used to obtain exact results.
- *Camera*: This menu contains sub menus for controlling the view camera.
 - *Front*: Rotates the view camera about the aimpoint to the front view (-Z axis)
 - *Back*: Rotates the view camera about the aimpoint to the back view (+Z axis)
 - *Left*: Rotates the view camera about the aimpoint to the left side view (+X axis)
 - *Right*: Rotates the view camera about the aimpoint to the right side view (-X axis)
 - *Top*: Rotates the view camera about the aimpoint to the top view (-Y axis)
 - *Bottom*: Rotates the view camera about the aimpoint to the bottom view (+Y axis)

- *Object View*: Copies the orientation (and other settings if included) from the selected object to the view camera
- *View to Object*: Stores the orientation (and other settings if the target is a camera) to the selected objects.
- *Perspective*: Toggles perspective and parallel camera projections
- *IPlane View*: Rotates the view camera so that it points orthogonally against the input plane
- *Auto Focus*: Moves, rotates and zooms the camera so that the selected objects fit into the view window suitably
- *Clip Focus*: Sets the near and far clipping distances to match the selected objects
- *Zoom In/Out*: Activates an interactor that LMB drags a zoom rectangle on a view window. Left-to-right drag zooms the view window inside the defined rectangle, right to left shrinks the current view to fit into the rectangle.
- *Tracking*: Switches camera object tracking on/off. More detailed control is available at the view property window.
- *Input Plane*:
 - *Front*: Rotates the input plane parallel to the XY plane about the grid origin
 - *Side*: Rotates the input plane parallel to the YZ plane about the grid origin
 - *Top*: Rotates the input plane parallel to the XZ plane about the grid origin
 - *Set Origin*: Activates an interactor that reads a LMB click (or drag-snap) from a view and moves the grid origin to the defined point.
 - *Match Surface*: Activates an interactor that reads a LMB click (or LMB-drag snap point) from a view and moves the grid origin to the surface point under the defined point. The input plane is rotated to fit the tangent plane of the underlying surface.
 - *Match 3 Points*: Reads three points from a view window. The points define the new input plane. The first point defines the new grid origin. The line between the two first points defines the new grid x axis.
 - *Lock to Camera*: Locks the input plane to view camera. The input plane is automatically kept orthogonal to the view camera direction.
 - *Grid Snap*: Turns grid snapping on/off.
 - *Grid Draw*: Turns grid drawing on/off
- *View Orientation*:
 - *Reset*: Initializes all view settings to the default front view (looking against z axis). The native view is reset back to the front view as well.
 - *Goto Native*: Sets the view window into the native state recorded into it.
 - *Current to Native*: Records the current view orientation and scale settings as the native state of the view.
 - *Object to Native*: Fetches orientation settings from a (camera) object and stores them as the native state.
- *World orientation*:
 - *From Object*: Fetches world coordinate orientation from the selected object. All numeric values will be measured along the object space directions.
 - *From Input Plane*: Fetches the world coordinate orientation from the current input plane orientation. Prior to using this tool, the input plane can be aligned to a surface tangent, an object space etc.

- *Set Origin*: Reads a mouse click (or snap point) and sets the origin of world coordinates there.
- *Reset*: Removes all world coordinate transformations.
- *Enter*:
 - *Compass Down Point*: Generates a left mouse button down event, which can be recorded to a macro. At macro execution, the click event will use the current mouse coordinates (not recording time coordinates).
 - *Compass Up Point*: Generates a left mouse button up event for macro recording.
- *Object Properties...*: Opens the property window
- *View Properties...*: Opens the view property window. With it, you can edit view window settings.

The Compass Menus

The idea of the compass menu is to provide a quick way to activate tools using the mouse. Use of normal pull-down and popup menus always requires careful aiming when an item should be picked. Compass menus are based on eight directions and two selection distances. These can be learned so that the hand remembers the motion, making the selection very quick.

The compass menu opens if you drag on a view window with the mouse, the right mouse button held down. The compass does not activate if a key modifier which selects another operation is held down - for example Shift-RMB-dragging just rotates the view.

The compass menu is context sensitive. The contents of it depends on key modifiers, object selection, object's edit state etc. The default view compass, shown above, opens if none of the selected objects specifies another kind of compass. You can also force the view compass by pressing the 'V' (=view) modifier key down before RMB-dragging.

Below, some general purpose compasses are described. Many object types and tools activate their own compass. Please check the object specific documentation for details.

Note: Modifier keys 'x' and 'z' are unused. They can be used in activation of user defined compasses. Most other keys are already configured as hotkeys that activate a tool, and their use as a compass modifier may lead to conflicting operations (compass activation performs another operation as a side effect).

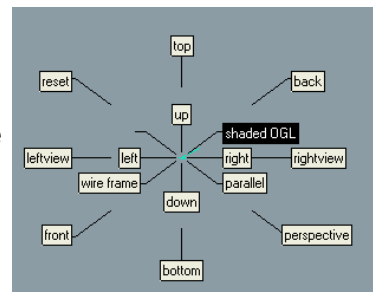
Navigation Compass

Description: Tools for changing the view orientation, the drawing mode and the projection

Activation: No compass modifiers active or 'V' key pressed

Items:

- *up*: Rotates view camera pitching angle upwards to the nearest 90 degree step
- *top*: Rotates the camera to the top view (direction against y axis)
- *shaded OGL*: Turns OpenGL accelerated drawing on
- *back*: Rotates the camera to the back view (direction along z axis)
- *right*: Rotates camera heading rightwards to the nearest 90 degree step
- *rightview*: Rotates the camera to the right side view (direction against x axis)



- *parallel*: Activates parallel view projection (view camera has infinitely large focal length)
- *perspective*: Activates perspective view projection
- *down*: Rotates the view camera pitching angle downwards to the nearest 90 degree step
- *bottom*: Rotates the camera to the bottom view (direction along y axis)
- *wire frame*: Selects wire frame drawing without OpenGL acceleration
- *front*: Rotates the camera to the front view (direction against z axis)
- *left*: Rotates the camera heading to the left, to the nearest 90 degree step
- *leftview*: Rotates the camera to the left side view (direction along x axis)
- *reset*: Resets the view back to its 'native' state

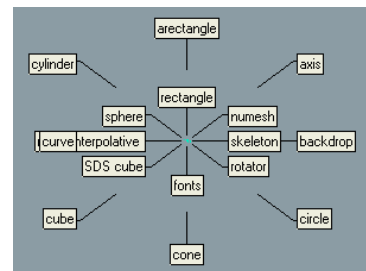
Creation Compass

Description: Frequently needed object geometry creation tools.

Activation: 'q' key pressed

Items:

- *rectangle*: A rectangular 4*4 cubic NURBS mesh
- *arectangle*: An analytic rectangle
- *numesh*: An 8*8 cubic NURBS mesh
- *axis*: A vertical line of 2 points (suitable as a rotation axis etc.)
- *skeleton*: Creates a skeleton object
- *backdrop*: Creates a camera backdrop plane. If a camera object is not selected, creates also a camera from the current view.
- *rotator*: Creates a rotator object.
- *circle*: A circular NURBS mesh
- *fonts*: Activates the font tool
- *cone*: An analytical cone
- *SDS cube*: A 2*2*2 subdivision surface cube
- *cube*: An analytic cube
- *NURBS interpolative*: Cubic NURBS curve creation through curve knot points
- *curve*: Cubic NURBS curve creation by control polygon points
- *sphere*: An analytic sphere
- *cylinder*: An analytic cylinder



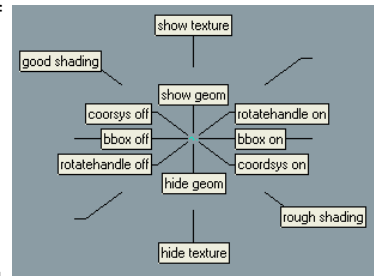
Wireframe Compass

Description: This compass controls various real time drawing options of geometric objects.

Activation: 'w' key pressed

Items:

- *show geom*: Makes hidden geometry visible
- *show texture*: Sets Property window/Wire tab/Texture quality of selected objects to 64*64 pixels.
- *rotatehandle on*: Makes object space rotate handle of selected objects visible. Note: View popup manu/Handles/ Rotate must be active, otherwise the handles are not visible.
- *bbox on*: Makes object space bounding box handle of selected objects visible. Note: View popup manu/Handles/ Bounding Box must be active, otherwise the handles are not visible.
- *coordsys on*: Makes object space axis handle (for translation and scaling) of selected objects visible. Note: View popup manu/Handles/Rotate must be active, otherwise the handles are not visible.
- *rough shading*: Sets Property window/Wire tab/Shaded quality of selected objects to the level 2.
- *hide geom*: Hides surfaces of selected objects. Possible sub objects or handles are not affected.
- *hide texture*: Sets Property window/Wire tab/Texture quality to 'None'
- *rotatehandle off*: Makes object space rotate handle of selected objects invisible.
- *bbox off*: Makes object space bounding box handle of selected objects invisible.
- *coordsys off*: Makes object space axis handle of selected objects invisible.
- *good shading*: Sets Property window/Wire tab/Shaded quality of selected objects to the level 4.



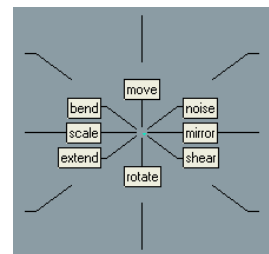
Transformation Compass

Description: Frequently needed transformation tools.

Activation: 'e' key pressed

Items:

- *move*: Activates the usual object translation tool.
- *noise*: Activates a transformation that moves points of selected objects randomly.
- *mirror*: Activates the tool which reflects the selected objects w.r.t. a given axis
- *shear*: Activates the shear (skew) tool
- *rotate*: Activates the rotate tool
- *extend*: Activates one-dimensional scaling along a given axis



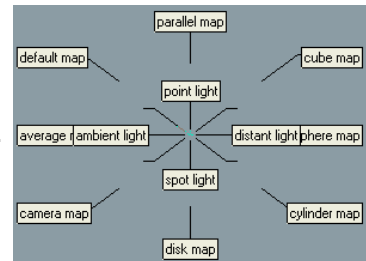
- *scale*: Activates the scale tool which allows rectangular scaling. Uniform Scaling can be obtained by pressing the Shift modifier down and moving the mouse diagonally.
- *bend*: Activates the tool for bending along a circle arc

Light Source and Material Mapping Compass

Description: Tools for creating light sources and material mapping objects.

Activation: 'y' key pressed

Items: Light sources at the near distance, mapping objects at the far distance.



- *point light*: Activates creation of the usual point light source
- *parallel map*: Activates creation of a parallel material mapping object
- *cube map*: Activates creation of a cube material mapping object
- *distant light*: Activates creation of a distant light source
- *sphere map*: Activates creation of a spherical material mapping object
- *cylinder map*: Activates creation of a cylindrical material mapping object
- *spot light*: Activates creation of a spot light
- *disk map*: Activates creation of a disk material mapping object
- *camera map*: Activates creation of a default material mapping object with the 'camera' coordinate option.
- *ambient light*: Adds an ambient light source into the scene
- *average map*: Creates a level object that computes a symmetric average of selected material mapping objects.
- *default map*: Activates creation of a default (no geometry included) material mapping object.

Navigation

You can navigate (move the view's camera in space) in several ways:

- Using the mouse with some modifier keys
- With keyboard shortcuts
- The view control window contains a set of navigation controls
- The compass menu includes navigation functions
- The popup menu provides a comprehensive set of frequently needed predefined orientations (front, side, top view etc.) and other similar controls
- With the numerical controls of the view property window

View navigation by the right mouse button:

Shift	Zoom (change view angle/zoom scale)
Apple	Pan (move camera)

Shift	Zoom (change view angle/zoom scale)
Alt	Rotate camera about aimpoint
Shift Apple	Move camera forward/backward
Shift Alt	Rotate in 45 degree steps
Apple Alt	Adjust camera banking
Shift Apple Alt	Adjust camera banking in 45 degree steps

View navigation by the middle mouse button:

Shift	Move camera forward/backward
Apple	Pan (move camera)
Alt	Rotate camera about viewpoint
Shift Alt	Rotate camera about viewpoint in 45 degree steps

View cruising with cursor keys:

Tab	View camera cruising on
Tab+Shift	Turbo boosted cruising
Tab+Up	Accelerate
Tab+Down	Decelerate
Tab+Left	Turn left
Tab+Right	Turn right
Tab+Alt+Up	Turn up
Tab+Alt+Down	Turn down
Tab+Alt+Left	Bank counterclockwise
Tab+Alt+Right	Bank clockwise
Tab+Apple+Up	Move up
Tab+Apple+Down	Move down
Tab+Apple+Left	Move left
Tab+Apple+Right	Move right
Tab+Left+Right	Turn towards horizontal flight

View Camera

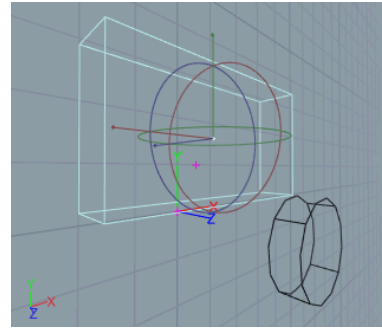
Every view window owns a private 'camera'. A view camera stores attributes which are familiar from real world camera: position, aiming direction, focal length etc. Thanks to this view camera concept, you can 'take pictures' i.e. render images on a view window without creating an actual camera object into the scene hierarchy. A view camera is identical with the camera objects in the scene hierarchy. You can easily assign all settings from a view camera to a camera object and back.

Input Plane

The input plane is a flat, infinite 2D plane in the 3D modeling space which you see through the view windows. The mouse pointer position is projected to the underlying input plane. This way 2D coordinates from the mouse become translated into 3D coordinates.

the input plane is, by default, connected to the view camera so that it always remains orthogonal to the camera. You can also unlock the input plane and model in a perspective manner. For example, you may want to view a room scene using a final camera view, place the input plane on the floor and move people inside the room along the floor to obtain an optimal composition.

The input plane becomes visible when you activate grid drawing. The input plane always matches the space division into grid snapping intervals. You can move and rotate the input plane using the view controls, numerically from the view property window or using the popup menu.



Perspective modeling on a vertical input plane

Most geometry creation and modify tools read their coordinate input from the input plane position. Certain tools (those using object handles) do not rely on the input plane, but project the mouse input to circular handles, axis systems etc. directly. For example, when you edit SDS object faces, input plane may be irrelevant.

View Control Bar

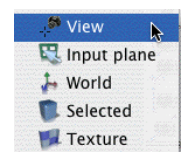
The View Control Bar is docked to the right side of the view window in the default configuration. You can open and close it from the 'View/ViewControls' pull down menu or using the F5 hotkey. If the contents of the view control bar do not fit inside the visible area, you can scroll it by dragging it with the right mouse button 'Apple' key held down, or with the middle mouse button.

You can open a popup menu from the left border of the view control bar. With the menu, you can turn icon labels on and off, and control the visibility of the view control groups.

- *Show labels*: Text labels under icons on/off.
- *Show Target Group*: The 'Target' group has only one selector gadget, which defines what the controls in the next 'Camera' group modify.
- *Show Camera Group*: This group contains the four controls for rotating, zooming, changing the distance and panning. By default, the target is the view's camera.
- *Show Interactor Group*: The 'interactor' group contains four tools which need mouse click input on the view window. The tools control the aimpoint and the input plane.
- *Show Orientation Group*: This group contains several tools for switching the orientation of various view items. For example 'Reset View' button is located here.
- *Show Canvas Group*: The Canvas group contains miscellaneous view controls. OpenGL, GDI drawing and Ray Trace buttons are located here as well.

The target selector gadget selects the modify target for the Camera group controls below the selector gadget. The following targets are available:

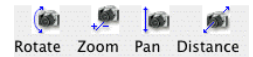
- View
- Input plane
- World orientation
- Selected objects
- Texture maps of selected objects



Camera Group

The Camera group contains four icons which control the orientation and position of the selected target. Use the icons by pressing the left mouse button down on top of an icon and moving the mouse LMB held down. The icons are (in up to down or left to right order):

- **Rotate:** Rotates the selected target.
- **Zoom:** Changes the camera scale / resizes the selected target.
- **Pan:** Moves the camera or another target sideways, up and down.
- **Distance:** Moves the camera forward/backward or another target closer/further away.



Interactor Group

The common nominator of the tools in the Interactor group is that after a tool has been selected, the program expects some mouse clicks (or LMB dragging) on the view window. The tools are:



- **Zoom:** This is a combined pan and zoom tool. To use it: Activate the tool. Then LMB drag a rectangle on a view window. Note how the dragged rectangle has automatically the proportions of the view window. After left-to-right dragging, the defined rectangle area is magnified to fill the entire view. Reversed dragging from right to left will shrink the current view area to fit the defined rectangle.
- **Aimpoint:** Sets the center point of the view. The new aimpoint position can be defined with a left mouse button click, but also snapping operations can be applied. Use this tool to center the view exactly to the desired point.
- **IP by 3P:** Reads three points and fits the input plane to match the defined plane. The line between the two first points will become the input plane's x axis. For example, you can fit the input plane to any triangle or rectangle by applying this tool: just activate the tool and then LMB drag three corner points with the depth modifier key ',' (comma) held down.
- **IP by N:** Fits the input plane to surface tangent. Activate the tool and LMB click over a surface. You may also LMB drag (preferably grid drawing on) to see how the input plane rotates; releasing the mouse button sets the input plane.

Orientation Group

The Orientation group contains tools which set predefined orientations for various view window related items. Most of these tool icons have also a side menu which opens with a LMB click. The side menus contain some additional variations of the tools.



Front: This tool puts the view camera to the front view orientation. The camera will look against z axis (heading 180 degrees, pitching zero). The camera is rotated to the front view about its aimpoint. Also the input plane is rotated so that its normal matches the z axis. Shift clicking the icon gives the opposite view (the back view).

The side menu has two additional versions: 'View' and 'Input Plane'. These tools rotate the view camera or the input plane only. Note, however, that all the three versions perform the same task when the camera and the input plane are locked together.

Side: The side view tool. Side view is the view from the left side of the object (along positive x axis). The side menu is similar to the front view above. Shift clicking the icon gives the opposite view (the right side view).

Top: The top view tool. View camera and input plane are rotated so that the camera looks down against y axis. See the front view tool above for the side menu explanations. Shift clicking the icon gives the opposite view (the bottom view).

Reset: Resets the view back to the 'Native state' that has been stored into it. This usually changes camera and input plane positions and orientation, as well as the zoom scale. The side menu has three entries:

- *View to default front view:* Total view camera reset. The native orientation may get corrupt, or otherwise to a badly defined state. This menu offers an easy way to put it back in order. Aimpoint and grid origin are set to absolute space origin, the camera takes a front view and the zoom scale is reset to the default value 1.0. The initialized values define the new native state of the view.
- *Reset world:* It is possible to define an additional global transformation for the whole scene hierarchy. This so called world transformation is useful when when the world coordinates must be aligned to a part of the scene hierarchy. For example, in a landscape model, the original world coordinates perhaps match the north-south and east-west directions. A building on the landscape seldom matches the same directions. When editing such an irregularly oriented building, it is helpful to set the world coordinates to match the walls. Numerical input for modeling tools becomes easy to interpret etc.
- *Set native orientation for the view:* Records the current state of the view window to be its native state. Next time the reset button is hit, the view returns back to the same state.

Fit: The autofocus tool. The view camera is aimed at the center of the selected objects and zoomed/moved so that the selected objects fill about 75% of the view window. Also the clipping range is adjusted so that all selected objects become visible.

Fetch: Fetches the view camera orientation from the selected object. The view will be moved to its position and rotated to match its object space. For example, you can take the view from a camera object or from a spot light this way. The side menu is the following:

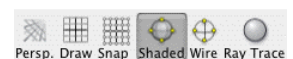
- *View to object:* Copies view orientation to selected objects. Usually this tool is used for recording a view to a camera object, but you can also set spot light orientation this way, for example.
- *Object to input plane:* Aligns the input plane to the selected object. Also the view camera follows if it is locked to input plane. It is easier to modify the object when mouse coordinates (and grid orientation) follow the object's orientation.
- *Object to world:* Aligns the world coordinate directions with the object space directions of the selected object.

Sync IP: Rotates the view camera so that it is orthogonal to the input plane. This tool may be useful when using a disconnected camera-input plane system. The attached menu is the following:

- *View orientation to input plane:* Same as above, but this time input plane is rotated, not the view camera.
- *Input plane orientation to world:* Aligns the world coordinates with the input plane. There are many tools which align the input plane to objects, surface tangent etc, which helps in accurate modeling. If usable reference coordinates are also needed and the input plane does not follow world's coordinate directions, you may have to match the world temporarily with the input plane.

Canvas Group

Persp: Sets perspective projection for the view on/off. The parallel camera projection is often more suitable for accurate modeling purposes.



Draw: Enables and disables snap grid drawing. A grid must be selected before it can be drawn.

Snap: Enables and disables grid snapping. A grid must be also assigned for the view, otherwise snapping does not work. You can assign the grid by drag and dropping it from the select window into the view window, or using the view property window.

OpenGL: Activates real time drawing via the OpenGL interface. By default, it renders shaded surfaces, but you can change the options from the view property window. OpenGL is a faster alternative than Windows GDI drawing if a modern graphics accelerator is available.

Windows: Selects Windows GDI drawing which is always in wireframe mode. GDI drawing is not hardware dependent and a safe alternative if OpenGL driver compatibility problems appear.

Ray Trace: Ray trace renders the view window.

View Properties

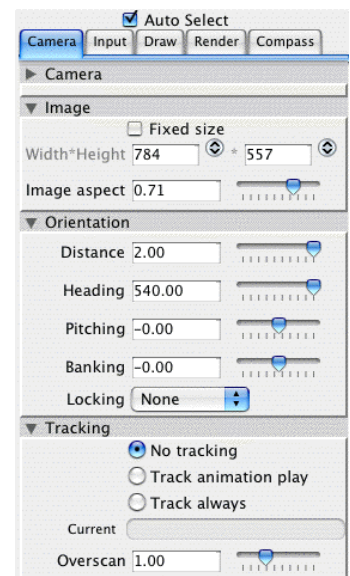
A view window is the most important element of a 3D program. Consequently, a large set of options is available for controlling its operation. These options are collected into the view property window.

The view property window opens from several places:

- From the view's popup menu 'View Properties...'
- From the pull down menu 'Windows/View properties'
- By double clicking a view model object in the select window

The Controls of the view property window are divided into several tabs, described below. The edited view model can be selected using the **Auto select** option; if the option is enabled, the view that has been clicked last (i.e. has the input focus) will become the editing target. Clearing the 'Auto select' option shows a list of all view models and you can activate the editing target by clicking an item on the list.

Camera frame: This is a standard camera object property gadget. Every view window contains an internal camera object and you can edit its properties here.



Fixed size: If set, the pixel resolution of the view window is determined by the Width and Height fields below this control. If cleared, the resolution is defined by a host container such as a floating window.

Width*Height: The pixel resolution of the view window. These fields control the resolution when the 'Fixed size' option is enabled.

Image aspect: The ratio $\text{height}(\text{pixels})/\text{width}(\text{pixels})$. Changing the value rescales height. Note that the aspect is computed from the plain pixel resolution, and does not take into account the pixel aspect ratio itself (non-square pixels).

Distance: The distance from the camera position to the aimpoint. Aimpoint is the origin of view navigation rotations and the focusing point for depth of field.

Heading: Camera's direction (axis projection) on horizon plane. Heading zero corresponds viewing along the positive z axis. The front view heading is 180 degrees.

Pitching: Camera angle with respect to the horizon plane. Front and side view pitching is thus zero, and top view pitching is 90 degrees.

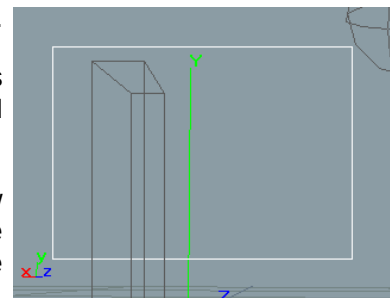
Banking: Camera's tilt angle.

Locking: View camera's orientation and other 3D space related properties can be locked here. For example, you may want to lock one view of a quadview system to display permanently the front view. Or, you can lock a certain carefully chosen camera setup on a raytrace preview window. This prevents accidental changes on the window after rendering tests when you return back editing the model.

- None - View camera can be modified without limitations
- All - Camera's direction, distance, position, lens opening angle etc. cannot be changed
- Direction - View camera keeps its direction but you can move it and zoom it

Tracking: Tracking controls define how the view camera should be connected to the currently active camera object in the scene hierarchy. The following alternatives are available:

- *No tracking:* View camera is independent of camera objects of the scene.
- *Track animation play:* View starts showing the current camera view as soon as the animation playback starts. Tracking is automatically ended when the animation play stops.
- *Track always:* Bi-directional tracking. View updates always to show changes in the current camera. Navigating or otherwise changing the view copies the view's camera state to the current camera object in the hierarchy (if such one exists).



Overscan tracking with value 0.75. The white rectangle defines the view area of the the tracked camera

Current: Displays the name of the tracked camera object.

Overscan: Overscan allows you to view slightly larger/smaller area than the actual tracked camera shows. For example, if the overscan is 0.75, the camera view will fill only 75 % of the view window. This may be useful when analyzing visibility of objects that spread their influence, for example glow. Value 1 gives an exact match.

Input

Grid frame includes grid snapping controls. Grid snapping is view specific: each view can use different grid settings.

Grid snap: Activates grid snapping. Mouse input will be rounded to grid units.

Grid draw: Activates grid drawing, which acts as a visual aid in modeling.

Current grid: Selects the grid which will be used for snapping.

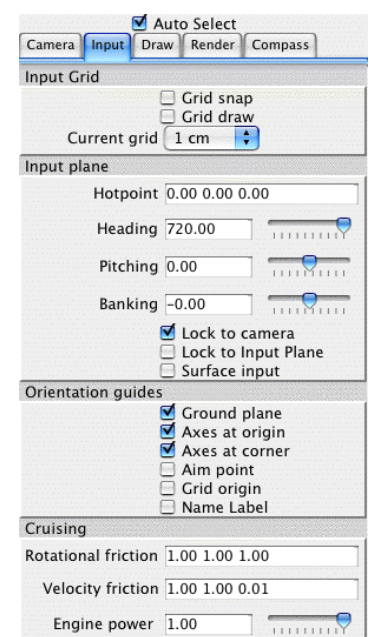
Input Plane is a two-dimensional infinite plane in the modeling space, projecting the mouse pointer movements to 3D coordinates.

Hotpoint: A point in space trough which the input plane travels. Defines also the grid origin point: grid snap intervals are computed starting from the hotpoint.

Heading: The projected direction of the input plane normal in the horizon (XZ) plane.

Pitching: The angle between the input plane normal and the horizon plane.

Banking: Tilt angle of the input plane. Banking rotates the grid around the input plane normal.



Lock to camera: If set, the input plane is automatically kept orthogonal to the view camera direction. When the user navigates on the view and rotates the view's camera, the input plane follows the rotation.

Lock to Input Plane: If set, selecting an input plane object or a level containing an input plane from the scene hierarchy automatically sets the input plane. The view updates automatically when the input plane is modified.

Surface input: If set, coordinates inputted from the mouse are projected to underlying object surfaces. For example, you can draw a path following a terrain surface.

Ground plane: Activates drawing of a horizon plane grid (XZ plane trough origin) in perspective view mode. The ground plane helps to keep track of directions when navigating on a view window. Ground plane is not drawn when using a parallel view projection.



Note

Do not mix the ground plane with visible grids. Ground plane has nothing to do with snapping.

Axes at origin: If set, a coordinate system is drawn at the origin point of the space. Each axis of the coordinate system shows a small label (X, Y or Z). The coordinate system has fixed 3D dimensions, so scaling the view window or moving camera further away from the origin makes it appear smaller, which is a useful visual aid in view navigation.

Axes at corner: Activates drawing of another, smaller direction indicator at the bottom left corner of the view window. This coordinate system is drawn at a fixed position and with a fixed size. Therefore, unlike the origin coordinate system, it remains always visible and available for orientation checking, regardless of view panning and zooming.

Aim point: Turns aimpoint drawing on/off. Aimpoint defines the position in space around which the camera navigates. Therefore, it always appears in the middle of the view window. The aimpoint marker has also 3D space defined size which indicates camera's distance from the aimpoint and zoom scale.

Grid origin: Draws a marker for the origin point of the input plane (sometimes called as the hotspot). This point is the base point for grid snapping. Most tools measure numerical input from the grid origin in the beginning of interaction. The grid origin marker has a 3D space defined size, meaning that it shrinks and expands when the view camera is zoomed or moved.

Name Label: If enabled, the name of the view model is drawn on the top left corner of the view window. This is useful for direction locked views, which can show the name of the projected direction (front/top/side). Also other kind of view windows can inform the user via the label option: a preview window can be labeled as 'Click me to render'.

The cruising controls are available for 'vehicle like' navigation in 3D space. Cruising behavior is controlled with a set of coefficients which describe the properties of surrounding space/matter (vacuum, air, water etc.) and the aerodynamical properties of the 'virtual vehicle'. The friction factors combine the space and aerodynamics properties. The higher a factor, the harder the surrounding matter resists the respective motion component. All these factors use a relative scale so that factor 1 gives a good default behavior.

Rotational friction: Defines how the surrounding space resists rotations around vehicle axes. The first factor is the friction of the rotation around vehicle's 'up' axis (heading), second factor is friction against pitching type rotation and the last factor is the banking friction.

Velocity friction: Friction factors for movements along the three main axes: sideways, up/down and forward/backward. A dart has factors like (1, 1, 0.1). A thin metal plate in water has two almost zero factors and one factor which is very high: (0.1, 0.1, 10). A car has factors like (sideways=5, up/down=100, forward=1) (actually 'down' friction should be even higher because the car cannot pass through the ground).

Engine power: Defines how quickly the vehicle accelerates when a navigation control is applied. The suitable value depends on the dimensions of the scene, speed of the computer, complexity of the scene etc. Too strong engine power makes navigation hard to control because changes happen too rapidly.

Draw

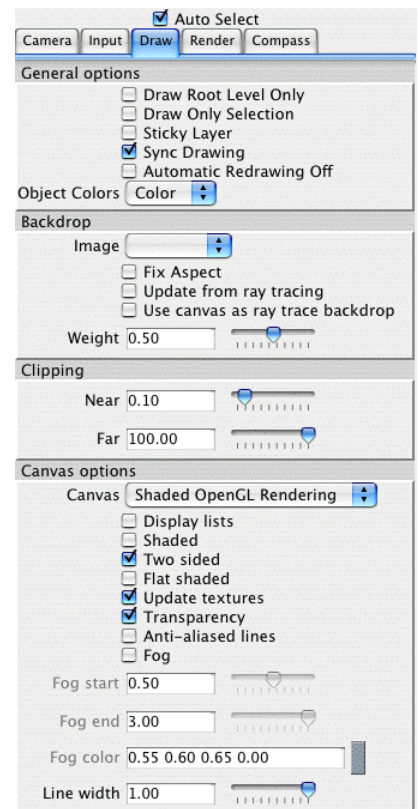
Settings of this tab control real time rendering.

Draw Root Level Only: When this option is enabled, only the part of the object hierarchy that is visible on the select window is drawn. You can define the appropriate branch of the object hierarchy by double clicking a level object with the Shift modifier key held down. A double click on the highest level moves the 'virtual root' one step higher.

Draw Only Selection: If set, only selected objects will be drawn. For example when point editing mesh objects, it may not be necessary to see other objects.

Sticky Layer: If set, the view window continues drawing the project which was active when the option was set. Normally the view windows draws the current project.

Sync Drawing: Instructs the program to finish the screen refresh before a new refresh cycle starts. Without the sync option, drawing interrupts much easier, which in some cases add flicker. Turning sync drawing off may be useful when working with complex scenes.



Automatic Redrawing Off: Normally view windows refresh automatically when the scene changes. This option turns automatic drawing off. The view will be redrawn only when the user changes the view orientation or selects Render/Wire, Ray Trace etc. from the view's popup menu (or uses another rendering command). The option is quite useful for ray tracing preview windows, which will then keep their latest rendered image as a reference while the user edits the scene.

Object Colors: Defines the color scheme for realtime rendering. The following alternatives are available:

- *Black:* Objects are drawn with two colors: one color for unselected objects and another highlight color for selected objects. These colors can be changed at Classes/Root/Wire Frame/Renderer/SelectedColor and UnSelectedColor.
- *Own:* All objects are drawn using the exact RGB values of object's diffuse color. No selection highlighting is used.
- *Class:* This alternative allows drawing by object type specific colors. The user can define an individual color for spheres, cubes etc. from the wire frame class attributes.
- *Color:* This is the default drawing alternative. It uses a modified version of object's diffuse color. A darkened version is used for an unselected and a lightened version for a selected object. The drawing system also verifies that no object is drawn with a color that is too close to the background color (making object invisible).

With the Backdrop options, the user can add an image such as a photograph to the background of the view window. The backdrop can be used as a modeling guide, or for image composition purposes.

Image: Selects the image object to be used as a backdrop. You can create image objects at the Image tab of the select window. You can also assign the backdrop by drag and dropping it from the select window into a view window. You can remove the backdrop by selecting the empty entry from the image gadget.

Fix Aspect: Normally the backdrop image is scaled to fit the view window dimensions exactly. If the Fix Aspect option is set, y scaling obey's image's original proportions and pixel aspect ratio. Usually this means that part of

the backdrop is clipped at the top and bottom parts of the view, or that the backdrop does not fill the entire view window in y direction.

Update from ray tracing: If set, the result of view ray tracing is automatically set as the view backdrop. A ray traced image may provide a useful reference for further modeling actions.

Use canvas as ray trace backdrop: The opposite of the previous option. The contents of the view window is used as a background image for ray trace rendering.

Weight: The visibility of the backdrop. Value zero makes backdrop invisible, value 1 fully visible. The intermediate values are useful for making a backdrop less distracting and object wireframes better visible over the background.

Clipping values define a visible distance region in front of the view camera. Only geometry within the region will be drawn. This sometimes helps to extract the relevant target area, which is under construction or needs modifications, from the rest of the (possibly complex) geometric data of the scene. Hardware accelerated rendering (OpenGL) often uses quite limited accuracy to store distances, for example 16 bit integers. In such cases, too wide clipping range makes z buffer inaccurate and leads to incorrect object visibility and other rendering errors.

Near: The start distance of clipping. By default, 0.1 meters.

Far: The end distance of clipping. By default, 100 meters.

Canvas options depend on the selected realtime rendering system. The following two alternatives are available by default:

Windows GDI

Double Buffer: Double buffering means that the screen is refreshed by drawing into a hidden buffer which is made visible after the image is ready. This reduces screen flicker.

Overlay: Allocates a separate mono color buffer to which editing handles are drawn. This reduces the amount of redrawn geometry data during object editing and hence speeds up screen updating.

Double Buffer Overlays: Activates double buffering for overlay drawing, reducing flicker in object handle drawing.

OpenGL

Display lists: Activates display list optimization. This may sometimes speed up drawing.

Shaded: Turns on drawing of filled, shaded surfaces.

Two sided: If cleared, only the polygons whose normal is facing towards a light are shaded. This speed up rendering but may create rendering errors (when using unclosed surfaces such as a rectangle).

Flat shaded: Turns off smooth normal interpolation.

Update textures: Controls texture updating. Turning updating off sometimes accelerates drawing considerably (when edition objects with high Texture quality and complex VSL materials).

Transparency: Enables alpha channel use in OpenGL drawing. Transparency is used extensively in making edit handles visible through shaded surfaces.

Anti-aliased lines: Wireframe anti-aliasing improves drawing quality but slows down screen refresh rate.

Fog, Fog start, Fog end, Fog color: Realtime fog effect. Rendered geometry will fade into the given fog color within a distance range defined by Fog start and Fog end. Fog effect gives an additional visual cue and helps in viewing complex wireframe models. The effect works best if the fog color is the same as the view's background color.

Line width: Defines wireframe line drawing width in pixels. If Anti-aliased lines is disabled, only integer values 1, 2, 3 etc. are significant. Enabling antialiasing increases steps in the width setting.

Render

Rendering Settings: Selects ray trace rendering settings for the view window.

Effect/Box: The post processing configuration that will be applied to all rendered image boxes.

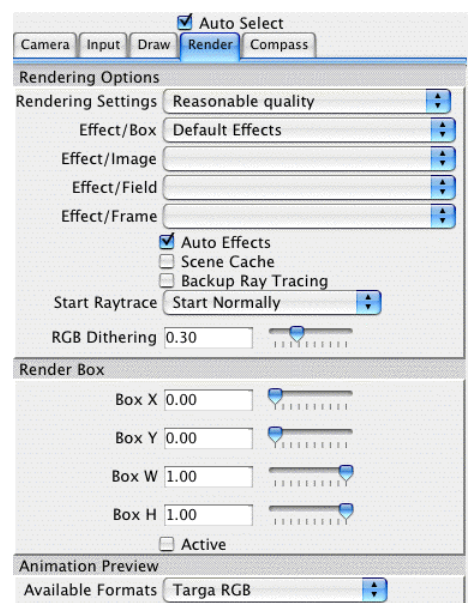
Effect/Image: These post effects are performed for complete images (not for single render boxes when using box rendering).

Effect/Field: Post effects that are applied after motion sample assembly to fields. Note that even when motion blur or field rendering is off, these effects are still applied to the finished image after box and image effects.

Effect/Frame: Post effects that are applied to complete frames after field assembly.

Auto Effects: Controls view's behavior in project loading. If the user loads a project which replaces the post effect library, but does not load view models (to keep the original GUI settings), the Auto Effect system assigns the first loaded post effect configuration to every view model, which has the Auto Effect option enabled. In other words, ray trace rendering of the view window will include the loaded post effects automatically. Auto Effect feature considers the Usage Recommendation of the loaded effects (per box, per image etc.)

Disabling Auto Effects may be desirable for special purpose view windows, for example to ensure maximally quick draft rendering without any post effects.



Scene cache: Activates a cache for ray trace rendering data. Optimization data from the previous ray tracing will not be deallocated after rendering. If only materials or the camera changes, new ray tracing starts much quicker (especially when using objects that require heavy preprocessing, such as dense SDS surfaces). The option adds memory consumption.

Backup Ray Tracing: When enabled, the deep channel data resulting from ray tracing is stored into a disk file. The data can be viewed later and the post effects can be rendered using it. This saves rendering time when adjusting or experimenting with the post effects, because ray tracing needs to be computed only once.



Note

If view window size or post effect configuration (and hence the required channel set) change, the backup file may become invalid. Post processing an invalid file may produce a black image.

Start Rendering: Selects the action which starts raytracing.

- *Start Normally* - Rendering starts when the user selects raytrace from the view's popup menu, hits the respective hotkey or clicks the raytrace view button.
- *Continuous Rendering* - Rendering starts automatically whenever the scene or the view orientation changes.

- **Click To Render** - A left mouse button click starts rendering. This option is very handy for a render preview window which is not used for object editing.

Box Order: Selects the order in which boxes are rendered

- *Row-Column* - Rendering starts from the left top edge and continues from left to right, row by row.
- *Scattered* - Boxes to be rendered are scattered all over the view.
- *Centered* - Rendering is started from the middle.
- *Inverted Row-Column* - Rendering is started from the right bottom edge.

RGB Dithering: The amount of view window specific RGB dithering that is applied when converting the 24 bit RGB data into the view to lower color depths. When using 24 bit display mode, the setting is unused.



Note

Render settings include a more accurate general dithering that works with all RGB bandwidths.

Box X: The left edge of the current render box measured in relative 0..1 coordinates. 0 = the left edge of the image, 0.5 = the middle point of the image etc.

Box Y: The top edge of the render box. 0=top edge of the image, 1=bottom edge of the image.

Box W: The relative width of the render box.

Box H: The relative height of the render box.

Active: Turns render box on/off. You can disable a render box to render a full view and then enable it again.

Animation Preview/Available Formats: Selects which format will be used to store the rendered images when view's popup menu Render/Animation Preview is used.

Compass

With the compass editor, you can configure a suitable set of compass menus.

Compass: The compass diagram shows the macros that have been bind to the selected compass. You can add items to the compass by drag and dropping named macros from the macro tab of the select window into the desired area of the compass. Items can be removed by applying the Delete popup menu on top of them.

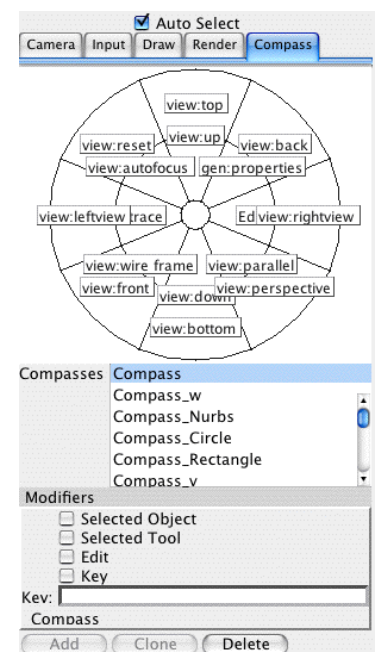
Compasses: The list of already existing compasses. You can click a compass item to activate it for editing.

The user can define several compasses per view window. The modifiers attached to compasses define which compass pops up when the user RMB drags with the mouse. The defined modifiers will be set when a new compass is created.

Selected Objects: Makes the compass object type sensitive. The object that is selected when this option is set will activate the compass.

Selected Tool: Makes the compass tool sensitive. The tool which has been activated when this option is set will activate the compass.

Edit: Makes the compass point edit mode sensitive. Compass will activate only if the selected object is in point edit mode.



Key: The compass is selected by a key modifier. Set this checkbox, activate the string gadget below and hit a key to be used as a modified. For example, view orientation compass appears if you press 'v' key while RMB dragging.

Add: Creates a new compass with the defined modifiers. If the modifier set conflicts with another compass, Add button remains disabled.

Clone: Copies the selected existing compass. Before a compass can be cloned, a different modifier set for the duplicate must be defined to ensure that compasses will not conflict.

Delete: Deletes the selected compass.



Chapter 2. BatchRen

Command line rendering utility

Description

Command line based batch rendering utility for rendering Realsoft 3D project files. Batch renderer needs less resources than the full Realsoft 3D program. Not only the program code is smaller but also the actual project consumes less memory space because only rendering specific data needs to be loaded into memory.

Batch renderer renders projects exactly the same way as File/Render window in Realsoft 3D. In fact, batchrenw and File/Render window are just two different interfaces to the Realsoft 3D rendering engine.

To use the batchren application, you need to create a project in Realsoft 3D, specify valid file rendering specifications via File/Render menu and then save the project.

Batch renderer uses the Realsoft 3D's distributed network renderer so all the options you specify in Realsoft 3D will also work in batch renderer.

You can pass any number of projects to batchren. The program loads the given projects one by one and renders each project separately.

Startup Options

The program supports a set of startup switches:

-file [file;file2...]	This option can be used for specifying one or more configuration projects. These projects are loaded with the actual project to be rendered. For example, the configuration project may define light sources, camera orientation and rendering settings allowing you to easily visualize iges, vrml and other files defining only geometric objects.
-plugins [foldername]	Specifies custom plug-in folder for the renderer. The default folder is Realsoft 3D's plugins folder.

-play	Renders all the frames of the project rather than the current frame.
-first [frame]	
-last [frame]	These options can be used for defining frame range for animation rendering. They have effect only if <i>-play</i> option is defined. To render a single frame, pass the same frame number to both options.
-out [filename]	Define file name for the image / animation to be rendered. This overrides the file name specified in the project's output settings.
-frset [objectname]	This option can be used for specifying the rendering specification object to be used for rendering the project. If not given, the currently selected configuration (the one shown by the File/Render window when the project was saved) is used. This option allows you render your projects with different resolution, quality etc. For example: <pre>batchren -frset PAL myproject.r3d</pre> <pre>batchren -frset NTSC myproject.r3d</pre>
-hosts ["host1;host2..."]	This option can be used for overriding the distribute network rendering servers specified in the rendering settings / hosts list of the project. For example to render 'test.r3d' by using three additional rendering servers (moon, mars and rserv.mydomain.com): <pre>batchren -hosts "moon;mars;rserv.mydomain.com" test.r3d</pre>
-width [xresol]	
-height [yresol]	These options override the image resolution defined by the project to be rendered. For example, to render 'myicon.r3d' project with 32 by 24 pixels: <pre>batchren -width 32 -height 24 myicon.r3d</pre>

Examples

```
batchren -file myconfig.r3d test1.igs test2.igs
```

```
batchren -file "flyaround.r3d;red.r3d;globalillum.r3d" car.r3d
```

```
batchren -play -first 10 -last 20 -out testanim.avi myproject.r3d
```

```
batchren -play -first 10 -last 10 myanim.r3d
```

```
batchren -out testimage myproject.r3d
```



Chapter 3. BatchRenW

Batch Rendering Utility, with graphical user interface

Description

This program is a rendering utility with a graphical user interface. To use the program, use the Browse button to add one or more project files to the project list and specify whether the project should be rendered as a still image or as an animation. When done, click Render All to get all projects in the project list rendered. You can also render a single project using the Render Selected button.

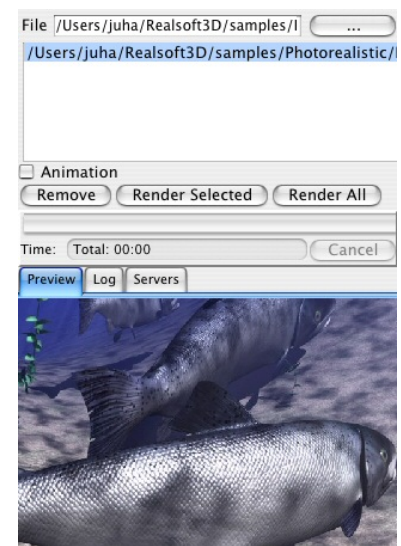
Due to dynamic and scalable nature of Realsoft 3D, batch renderer needs significantly less resources than the full Realsoft 3D program. Not only the batch renderer application is smaller but also the actual project eats less resources because only rendering specific data is loaded into memory.

Using BatchRenW

Batch renderer renders projects exactly the same way as File/Render window in Realsoft 3D.

To use the batchrenw application, create one or more projects in Realsoft 3D, specify valid file rendering specifications via File/Render menu for them and then save the projects. In batch renderer, add the projects to the project list and click Render All. Batch renderer uses the Realsoft 3D's distributed network renderer so all the options you specify in Realsoft 3D will also work in BatchRenW.

The Servers tab has Autodetect check box. When set, the program will use all available rendering servers for rendering. If not set, then the rendering settings saved with projects determine the servers.





Chapter 4. RendD

Network Rendering Daemon

Description

RendD is the actual rendering service for the distributed rendering system. It can be started directly or through RendW program.

RendD program must be running on every workstation which is acting as a rendering server. The program listens incoming rendering requests and starts rendering threads to take advantage of the processor power available in the workstation.

The program can be used for distributing rendering over any TCP/IP based network, such as the Internet.

In addition to local threads, RendD can distribute rendering to other rendering daemons running on other workstations. This means that hierarchical rendering farms can be configured.

In distributed network rendering, all data of the rendered project needs to be sent to all rendering servers. Transmitting data becomes often the bottleneck especially when one needs to distribute rendering over slow networks (which may be the case when distributing over the Internet). By using hierarchical rendering networks, the client sends projects only to a small number of servers, which then distribute the task further to specified sub daemons of a fast LAN etc.

Two methods exist for setting up hierarchical rendering farms. Sub rendering daemons can be specified 'manually' as startup parameters. Another alternative is to the Automatic Sub Servers option, which automatically distributes rendering to the workstations available in LAN or registered to it. Automatic approach has some advantages, because it makes the configuration of the rendering farm flexible - it can change any time.

Startup Switches

-noproxy

Turns off image map caching to the local system. The proxy option is active by default. Proxy system fetches referenced texture maps (and other required image files), which cannot be directly found from the given search

paths. The files are transferred in the native compressed form and saved to a temporary subfolder in the user's document folder. This has certain advantages:

- If a cached texture file is referenced again in a later rendering session, it can be loaded quicker from the local hard disk without any increase in network traffic.
- Proxy reduces network traffic significantly in hierarchical rendering configurations. This can have a dramatic impact on rendering speed when some of the network connections are slow. It is much quicker to move the textures only once over a slow internet connection to a local proxy server which then distributes them further in a fast local network than let all the servers to access the textures directly over the slow internet connection.

Proxy feature checks file sizes and dates and updates the cache automatically if necessary.



Note

To utilize proxy feature maximally, remove all search paths from the rendering daemon. Then all textures will end up into the local temp folders.

-proxycleanup

This option controls the proxy cleanup system. The temp folder is checked frequently and files older than the given amount of days are removed. The default value is 30 - the average lifetime of image files in temp folder is one month.



Note

Do not place important files into the temp folder! Proxy cleanup will delete all outdated files.

-threads [num]

Specifies the number of rendering threads spawned for each rendering request. The default value is the number of processors in the workstation.

-clients [num]

Specifies the maximal amount of clients, which will be served simultaneously. The default amount 1 is suitable for single user network rendering. In a multi user network, the daemon stops spawning new rendering threads soon as the defined client count is reached. The clients attempting to get rendering service from the fully occupied daemon will receive a failure code (which normally just eliminates the server from that particular rendering job but does not stop rendering, provided that other vacant rendering threads will be available).

-help

Prints out brief instructions for use.

-plugins [foldername]

Specifies a custom plug-in folder for the rendering daemon. The default folder is 'Plugins' (in the program's home folder).

-xboxcount [num] -yboxcount [num]

In order to distribute a rendering task, rendering daemons must subdivide the image to a number of sub images. Each sub image is then rendered by a separate local thread or a sub daemon. When all sub images are ready, they are combined and returned back to the client. The box count parameters can be passed to override the automatic box count detection system.

Note that the sub image count should always be higher than the number of rendering threads and sub daemons available. For example, if -threads is 2 and 4 external workstations are used as sub daemons, then the recommended

box count is 6 or more. A good choice could be 6 multiplied by 4, because rendering speed varies over the image area (an empty black corner renders instantaneously whereas a sub image including a glass object is much slower to render). A suitably high box count adapts better to such variations and keeps all processors busy. On the other hand, there is some overhead in managing the sub images - a very high number of boxes can start slowing down rendering.

If the user does not define any box count, the program estimates an optimal amount from the number of rendering threads and sub servers.

-paths [path1;path2;..]

This option allows you to define daemon side texture paths, which are used for resolving texture names.



Note

When using the texture proxy option, the texture paths should not refer to folders in external hard drives. An image is fetched to the proxy cache only if it is not found otherwise. Therefore, too extensive search paths may prevent proxy operations. The result is increase in network traffic. Search paths may safely include local texture libraries. For example, the local temp folder is automatically included in search paths.

-host [hostname] -port [port number]

These switches specify the host name and the port number used for connecting to this daemon. The default host name is the hostname of the workstation. However, sometimes you may need to override the default name, for example to make the daemon accessible through the internet.

The default port number is 10050. You may need to override the default port number in case the port is already used for something else.

-autodet

Enables the Automatic Sub Servers feature, which is useful when configuring hierarchical rendering farms. When enabled, the rendering daemon searches the network and automatically connects to all sub servers available.

-mem [bytes]

This switch is considered when the daemon is accessed by a client using Automatic Network Rendering. The value specifies how much memory this system can consume in rendering. All project files, which need more memory, will not be sent to the server. The default value for this parameter is the amount of the physical memory.

Examples

Use a workstation as a rendering server

If you just want to use a workstation as a rendering server, start RendD without any parameters:

```
RendD
```

The program will detect the number of processors in the system and automatically configure itself to take advantage of all the processing power.

Set up a rendering farm

Let's assume you have a network consisting of five workstations and you want to let your employee to access them through the internet. To minimize overhead caused by data transfers, you need to set up one root daemon and four slave daemons. Render farm clients connect only to the root daemon, which then distributes the rendering requests to the other four workstations. It is much faster to send the data to the workstations through the LAN than by sending them through the Internet.

To configure this, start RendD on the four slave workstations without any parameters:

RendD

Start the root daemon by specifying the host names of the four slaves as startup parameters:

RendD slave1 slave2 slave3 slave4



Tip

In Windows XP, you can find the host name of a computer from Control Panel/System/Computer Name tab. You can use the full computer name defined there as the host name.

Use only one processor for rendering

If you have a dual processor system in the rendering network and you want to use only one of the processors for rendering, start the network rendering daemon as follows:

RendD -threads 1

Create a hierarchical rendering farm

Let's imagine you have a LAN consisting of large number of workstations. The LAN configuration or availability of its computers for rendering purposes changes constantly. In such a case, it is best to use the Automatic Sub Servers option.

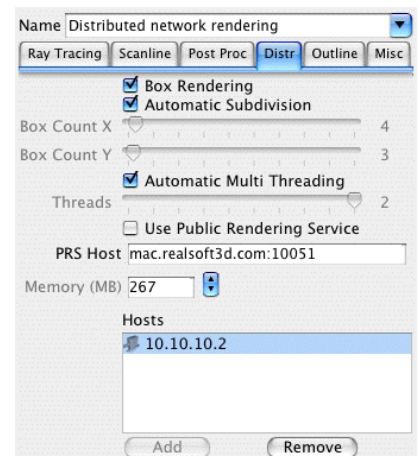
Let's assume one workstation 'host5' can be constantly available for rendering. Start the network rendering daemon in it as follows:

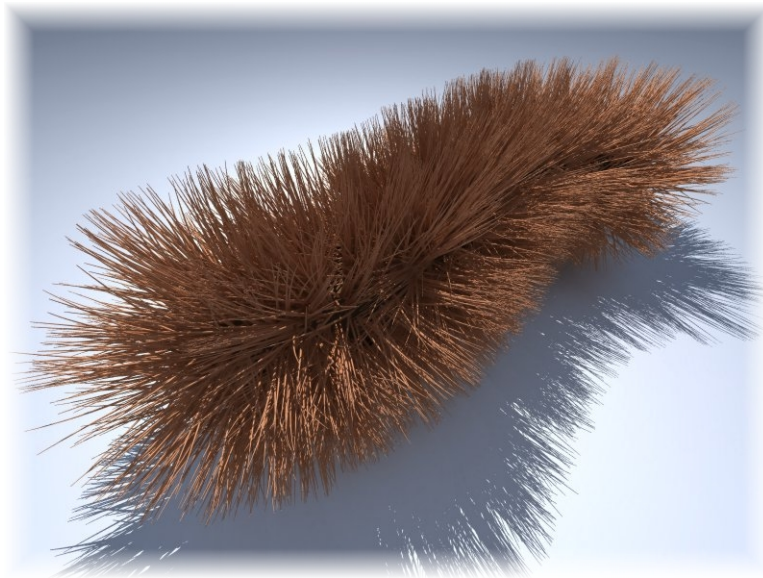
RendD -autodet

Start the render daemon in other workstations as follows:

RendD

In the render settings of the client application, for example a view window of Realsoft 3D, enter the host name 'host5' into the Hosts list. Do not use the Automatic Network Rendering option, because the program will connect to the servers both directly and via host5.





Chapter 5. RendW

Network Rendering Daemon with a graphical user interface

Description

RendW is a graphical front end to the network rendering daemon RendD. RendW can start the RendD daemon any time, so that other users can start using your workstation, and stop it in case you need all the resources for yourself.

Pull-down Menus

When RendW is started, it creates a task bar icon, through which you can control various rendering options (Windows and Mac OSX versions only). There is a pop-up menu associated with the task bar icon, containing the following items:

Open	Open the window for controlling the daemon's options
On-line	Start the network rendering daemon (RendD) and allow network users to access the workstation for rendering.
Off-line	Stop the network rendering daemon. The workstation no longer provides network rendering services. Ability to enable/disable the rendering daemon is particularly useful when the rendering resources are shared using automatic network rendering. Users can add and remove resources to the rendering network in an easy, flexible way.
Shutdown	Shuts the RendW program down completely, removing the task bar icon and stopping the network rendering daemon.

Startup Options

The program accepts the same startup parameters as the RendD daemon. See RendD documentation for more information about the RendD specific startup options.

In addition to this, the program defines the following startup options:

-hide

The Control Window

When the task bar icon 'Open' is selected, the window shown on the right is opened. The options of the window correspond the startup switches of the RendD rendering daemon.

Enable

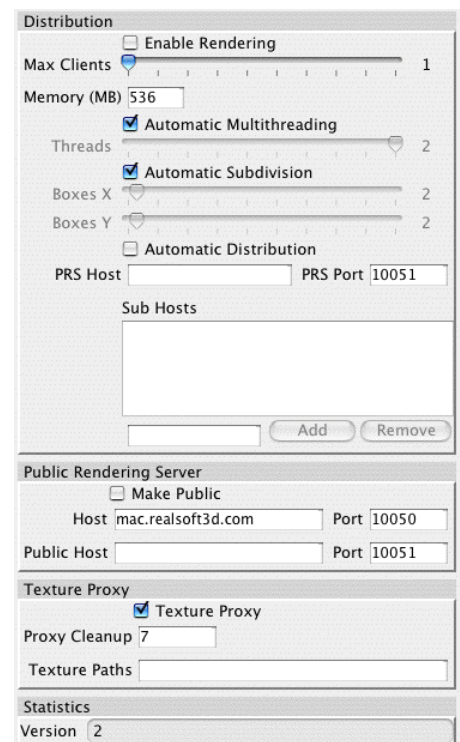
Starts / stops the rendering daemon (RendD) with the specified options. This option can also be controlled via the On-line and Off-line pop-up menus associated with the task bar icon.

When enabled, network users can use the workstation for rendering.

Max Clients

Specifies the maximal amount of clients, which will be served simultaneously. Under normal circumstances (i.e. when the daemon is configured to use all available processors and sub servers), this control should have value 1. Then only one client will be served at a time. Providing rendering services to several clients simultaneously can cause virtual memory swapping, which slows down rendering. Powerful multi processor systems with a large memory capacity can naturally serve several clients. For example: a quad processor system with 8 gigabytes of RAM in a network of multiple users, who constantly need rendering power, could be configured as follows:

- Max Clients = 4 (or more, up to the number of network users)
- Automatic multi threading = off
- Threads = 1



With these settings, the daemon will give each client only one rendering thread. One client cannot use all resources of the workstation and therefore several clients can be served efficiently.

Automatic Subdivision

Enables automatic image sub division. If checked, the program attempts to find an optimal box count based on the image resolution and the number of threads and sub daemons in external workstations.

BoxCountX / BoxCountY

If automatic subdivision is disabled, these fields determine the number of sub images generated to distribute rendering.

Automatic Multithreading

Enables automatic thread detection to take advantage of multiple processors available in the workstation (SMP). If checked, the number of rendering threads, which are started to serve each client corresponds to the number of processors. In other words, a client gets the total rendering power of the server computer.

Threads

The number of threads spawned for rendering. This option can be controlled only when *Automatic multi threading* option is not set.

Automatic Sub Servers

This option is needed only when setting up hierarchical rendering farms. If checked, the rendering daemon searches the network and joins the other servers it finds to the rendering farm. For example: a client connects to only one rendering daemon via internet. The daemon then redistributes rendering to its own sub servers using this option. The actual client will not know how many servers eventually render the image.



Note

You should never set the Automatic Sub Servers option for multiple rendering daemons in a same local network. Similarly, the rendering client and the rendering daemon running on a same LAN should not use the automatic configuration feature simultaneously.

Memory

Specifies the maximal amount of memory (in megabytes), which the computer system is allowed to use for rendering. This option is used only when the client uses the Automatic Network Rendering option. Clients won't send projects to servers, which don't have enough memory to render the scene or whose owner does not want to use more memory than a certain limit. By default, the value is set to the amount of physical memory.

Add / Remove

Add and Remove buttons below the host list allows you to add and remove host names manually. These fields are disabled if the Automatic Sub Servers option is active.

Disabled

If the host list is configured manually (Automatic Sub Servers off), you can select a host from the list and disable it temporarily using this checkbox.

Host / Port

These fields specify the host name and port number through which clients can contact the daemon.

Texture Proxy

Turns on image map caching to the local system. The proxy option is active by default and should be used by default.

Proxy Cleanup

This option controls the proxy cleanup system. Textures older than the given amount of days are removed. Cached textures are placed into temp folder.



Note

Do not place important files into temp folder! Cleanup will remove them.

Texture Paths

This field specifies search paths for textures. This field is only needed when texture proxy is not used. More than one search paths can be defined by separating them using a semicolon.

Debug

Enables printing of additional diagnostic information into the server log tab.



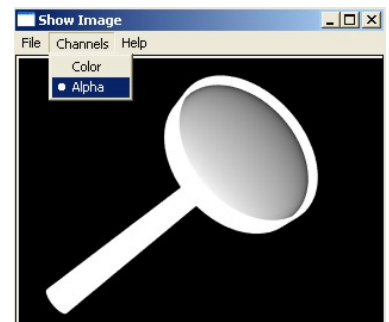
Chapter 6. ShowImg

Image Viewing Utility

This program can show any image format supported by Realsoft 3D. It can also compose and show temporary image boxes generated by distributed network rendering.

Pull-down Menus

File/Open Rendering Boxes	This menu allows you to select one of the sub images generated by distributed renderer. All the other sub images are automatically loaded and shown. Missing sub images are shown as black boxes. This function is useful for viewing sub images generated by distributed rendering system. If you leave the programming running, it automatically reloads the images. This allows you to monitor how network rendering progresses.
File/Exit	Exit ShowImg.
Channels	Realsoft 3D images can consists of any number of channels, such as Color, Alpha, Depth, to name a few. This menu can be used for selecting the channel to be shown.
Help/About	Open About dialog, showing the version number of the program.
Help/Manual	Open this reference page.



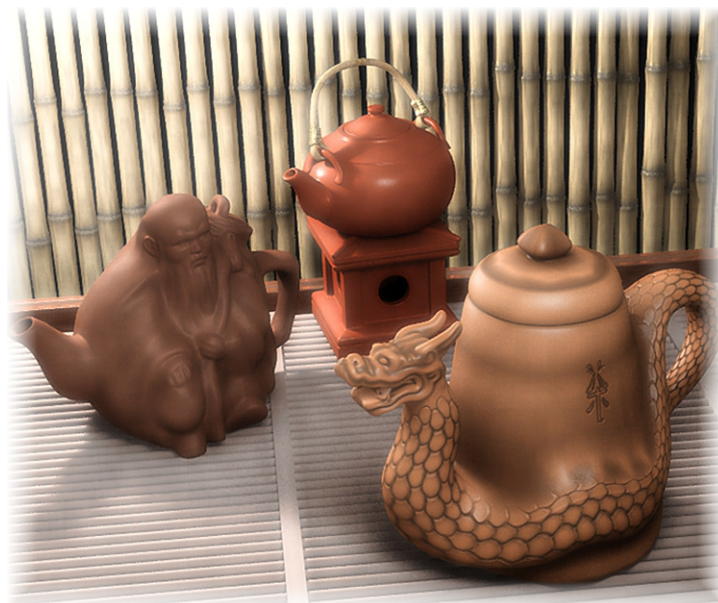
Startup Parameters

Showimg accepts number of startup parameters.

-file [filename]	Specifies the default name of the image to be shown.
-boxes	If set, the program assumes the given file name refers to set of sub images which should be composed to complete image, if possible. For example, network renderer subdivides image to a number of sub images to get the rendering job distributed to several threads / workstations. In case of system crash, this switch can be used for asking the program to read set of sub images rather than single image file.
-plugins	Specifies custom plug-in folder from which the plug-in modules should be read. The default folder is 'Plugins' (in program's home folder).

-boxprefix

This specifies a string to be used as a prefix for constructing box names. The default prefix corresponds to one used by Realsoft 3D's distributed rendering system.



Chapter 7. StartOpts

Startup option utility

Description

StartOpts program is a utility for defining startup options for Realsoft 3D.

Startup options

You can change the following startup options:

Texture Paths

This field specifies search paths for textures. More than one search path can be defined by separating them using a semicolon.

Plugin Path

You can enter a path where Realsoft 3D searches for plugins.

Home Path

You can enter the home path for Realsoft 3D.

Network Home Path

You can enter the network home path for Realsoft 3D, if Realsoft 3D is installed on a network drive.

Language

You can choose between the installed languages for the graphical user interface of Realsoft 3D.

The screenshot shows the StartOpts utility window with the following sections:

- Paths**
 - Texture paths: textures
 - Plugin path: plugins
 - Home path:
 - Nethome path:
- GUI**
 - Icons: Chrome (dropdown menu)
 - ☐ Don't load GUI
 - ☐ Force to screen
 - ☐ Balanced tree
 - Conflict color: 1.0,0.0
 - ☐ Html Help
- Network Rendering**
 - Render port: 10050
 - PrsHost:
 - PrsPort: 10051
- Startup**
 - Startup file: startup.r3d
 - Script:
- Debug**
 - ☐ Debug

Icons

You can choose between the installed icons for the graphical user interface of Realsoft 3D. Icons are installed into Realsoft 3D\r3icon folder.

Don't load GUI

Specifies that the layout of the graphical user interface is not to be loaded if Realsoft 3D opens a project file. This can be useful if you are loading other people's projects which contain customized user interfaces.

Force to screen

This option moves all open windows into the visible screen area so that the user can see them. This is useful if you are loading other people's projects which use a different monitor resolution or multi monitor setups and the windows are displayed outside of your screen's bounds.

Balanced tree

Specifies that hierarchical windows are displayed as balanced trees.

Conflict color

Specifies the conflict color for gadgets. The conflict color is displayed if e.g. conflicting options are specified in a gadget. The conflict color is specified as r,g,b with float numbers. To change it e.g. to green, enter 0,1.0,0 in this field.

Html help

If enabled, the program will use the documentation in html form instead of the native format. When using this option, the html manual must be installed.

Render port

Specifies the port number that the rendering server uses.

Startup file

Specifies the startup file to be used for loading the initial configuration for Realsoft 3D.

Script

Specifies the startup script to be executed when Realsoft 3D is loaded.

Debug

If checked, Realsoft 3D reports all undefined internal attributes and methods. This can be useful to track down old plugins, for example.



Chapter 8. RsView

Scene Viewer Utility

Description

This program can be used for viewing .R3D project files. It allows the user to navigate in the loaded scene using the Walk-Thru tool. The utility uses collision detection for floor and wall detection, allowing the user to move in the scene easily.

The tool can also run animations included with the scene. For example, one can create a simple key frame animation which opens a door. When the user looks at such a door in RsView and presses the space bar, the animation will be executed. When the user hits the space bar the second time, the animation will be run backwards and the door will close.

This program is freely distributable. One can create a .r3d project with Realsoft 3D license and send the project along with this utility to his/her customer, for example.

Pull-down Menus

File/Open	Open the scene to be explored.
File/Exit	Exit the program.
Channels	A scene to be explored.
Help/About	Open About dialog, showing the version number of the program.
Help/Manual	Open this reference page.

Startup Parameters

RsView can be called from a command line window with one or more project file names:

```
RsView myhouse.r3d mylightning.r3d ..
```

Projects are all loaded in and combined for browsing.

In addition to this, the program accepts the following startup options:

-file [filename]	Specifies the default name of the project to be loaded.
-plugins	Specifies custom plug-in folder from which the plug-in modules should be read. The default folder is 'Plugins' (in program's home folder).

Index

Symbols

1D Particles, 38
2D Particles, 39
3D Measure, 30
3D Measure Tool, 207
3D Painting Circle Tool, 175
3D Painting Curve Tool, 175
3D Painting Polygon Tool, 176
3D Painting Rectangle Tool, 177
3D Painting Selected Tool, 177
3D Painting Tool, 174
3D Particles, 39

, 115, 205, 205

A

Add Curve Tool, 236
Advanced Selection Window, 282
Advanced VSL Controls, 88
Aeroplane Banking, 40
Aimpoint, 16
Airbrush Tool, 174
Ambient Light, 62
Ambient Light Tool, 216
Ambient Occlusion, 114
Ambient Tool, 216
Analytic Rectangle Tool, 180
Analytic Tools, 179
Analytic/Rectangle Tool, 180
Anchor, 83
Angle Measure, 9
Angle Measure Tool, 207
Animateable Attributes, 289
Animated Image Object, 86
Animated Texture, 105
Animation Object, 145
Animation Properties, 289
Animation Settings Window, 283
Animation Window, 284
Antialiasing, 140
Antialiasing Trigger Channels, 141
Assign View to Selected Objects, 209
Assign View Tool, 209
Atmosphere Object, 9
Atmosphere Tool, 196
Attributes Window, 286
Auto Map Tool, 206
Autofocus Tool, 209
Automap Tool, 207
Average Map, 56

Average Map Tool, 218
Average Mapping Tool, 218
Average of Channels, 112
Average of Sub Channels, 112
AVI Output, 120
Axis, 148

B

Backdrop Object, 9
Backdrop Tool, 196
Base, 15
Base Index, 106
Batch Rendering Utility, 337
BatchRen, 335
Begin Illumination, 99
Bend Constructor, 74
Bend Tool, 263
Bindings, 116
Bleed Option, 105
Blur Effect, 127
BMP Output, 123
Boolean Operation, 29
Boolean Tool, 184
Boundary Surface Solid, 29
Box Count, 143
Box Rendering, 7, 143
Branch, 10
Branch Tool, 244
Break Curve Tool, 240
Break Mesh Tool, 237
Building Object, 14
Building Tool, 185
Building Tools, 185
Built-in JavaScript Classes, 148
Bump Height, 105
Bump Map Channel, 105
Bump Map Wizard, 117
Bump Mapping, 105
Bump Smoothing, 105
Bump VSL Object, 105

C

Call VSL Object, 106
Camera Object, 15
Camera Tool, 196
Camera Tracking Tool, 210
Carriage Object, 75
Carriage Tool, 189
Caustics, 145
Caustics Map, 63
Cell VSL Object, 106
Center of Gravity, 68
Center of Gravity Tool, 262
Center Pivot Tool, 222
Central Glow, 124
Channel Object, 1

Channels Tab, 310
 Choreography Classes, 2
 Choreography Object, 17
 Choreography Tool, 196
 Choreography Window, 286
 Chrome Wizard, 117
 Cineon Output, 120
 Circle, 20
 Circular Trim Curve Tool, 227
 Class Options, 298
 Client Window, 279
 Clip Map Wizard, 117
 Close Curve Tool, 240
 CloseU Tool, 237
 CloseV Tool, 237
 CoBind, 189
 COG Tool, 262
 Collision Accuracy, 70
 Collision Detection, 70
 Color of Surface, 96
 Color Square, 279
 Color Wheel, 291
 Color Window , 279, 291
 Color Wizard, 117
 Comment VSL Object, 107
 Common Properties, 95
 Compass Menu, 319
 Concatenate Curves Tool, 240
 Concatenate Meshes Tool, 237
 Conditional VSL Expressions, 109
 Cone, 17
 Constant VSL Object, 107
 Constraint Object, 17
 Constraint Tool, 197
 Construction Stack, 2
 Construction Tool, 189
 Constructors, 74
 Container Window, 279
 Control Bar, 314
 Coordinate System, 149
 Copy VSL Object, 108
 Crease Tool, 251
 Create Subdivision Object Tool, 252
 Creator Object, 18
 Cube, 19
 Cube Map, 56
 Cube Map Tool, 218
 Cube Mapping Tool, 218
 Curve Editor, 292
 Curve Gadget, 292
 Curve Manager, 19
 Curve Painting Tool, 175
 Curve Quality, 73
 Curve Types, 108
 Curve VSL Object, 108
 Custom Channel, 1

Custom Channel Object, 1
 Cut Cone, 19
 Cut Pyramid, 20
 Cycle, 280
 Cylinder, 20
 Cylinder Map, 57
 Cylinder Map Tool, 218
 Cylinder Mapping Tool, 218

D

Default Map, 57
 Default Map Tool, 219
 Default Mapping Tool, 219
 Define Input Plane by 3 Points, 210
 Define Input Plane by Surface Normal, 210
 DefineUV Tool, 237
 Deformation Tool, 200
 Delete Curve Points, 240
 Delete Points from Mesh, 238
 Delete Points Tool, 222
 Density, 69
 Density Wizard, 117
 Depth of Field, 17
 Depth Of Field Effect, 128
 Derive VSL Object, 108
 Destroy Faces Tool, 253
 Detach Tool, 206
 Digits of Texture Index, 106
 Disable VSL object, 92
 Disk Map, 57
 Disk Map Tool, 219
 Disk Mapping Tool, 219
 Displace Tool, 231
 Displacement, 36
 Displacer, 74
 Distant Light, 62
 Distant Light Tool, 217
 Distribute Points Tool, 240
 Distribute Tool, 212
 Distributed Rendering, 142
 Dithering, 142
 Dithering Effect, 129
 Drag Object, 75
 Duplicate and Mirror Tool, 253
 Duplicate Faces Tool, 253
 Duplicate Selected Edges Tool, 253

E

Eaves, 43
 Edge Filter, 130
 Edge Handles, 49
 Edge Rounding Tool, 253
 Edit NURBS Mesh Selector, 238
 Edit Objects Tool, 222
 Elasticity, 68
 Electric Charge, 68

Elevate Degree Tool, 240
ElevU Tool, 238
ElevV Tool, 238
Ellipse, 20
Ellipsoid, 20
Ellipsoid Segment, 21
Else VSL Object, 109
Emitter, 38
End Condition, 33
End Illumination, 100
Evaluate with Normal, 73
Explode at Impact, 71
Explode Time, 71
Explosion Pressure, 71
Explosion Resolution, 71
Extend Tool, 263
Extract Curves Tool, 237
Extrude Curve Tool, 241
Extrude NURBS Tool, 231
Extrusion Manager, 21

F

F-Stop, 16
Face Handles, 49
Faces Tool, 254
Fading, 55
Falloff, 60
Fan, 21
Fan Tool, 262
Fetch View Tool, 209
FFMPEG Output, 121
Fidelity, 83
Field Evaluator, 22
File Instance Object, 22
File Instance Tool, 197
File Render Configuration, 5
File Rendering Configurations Tab, 312
File Rendering Window, 294
Fill Holes Tool, 254
Fillet Tool, 231, 238
Film Sensitivity, 15
Finite, 56
Fit Tool, 209
Flash Light, 16
Flatten Tool, 255
FLC Output, 121
Flip Faces Tool, 255
Float, 166
Float Window, 281
Fluid Dynamics, 70
Fluid Friction, 69
Fluid Pressure, 69
Fluid Symmetricity, 69
Fluid Velocity, 69
Focal Length, 16
Focus, 16

Fog Effect, 131
Fog Wizard, 117
Font Manager Object, 23
Font Tool, 232
Foot Tool, 207
Force to screen, 349
Forced Visibility, 74
Format String, 106
Fractal Dimension, 110
Fractal Plant, 46
Frame, 280
Frame Command, 8
FrameBlur, 131
Freehand Knife Tool, 255
Frequency, 146
Front View Tool, 209

G

General VSL Options, 90
Geometric Objects Tab, 300
Geometry Manager, 207, 277
GIF Output, 121
Glass Wizard, 118
Glow Effect, 132
Glow Wizard, 118
Gradient Curve, 108
Gradient in Texture Mapping, 115
Gravity, 70
Gravity Object, 24
Gravity Tool, 263
Grid Draw Tool, 209
Grid Drawing, 85
Grid Object, 85
Grid Snap Tool, 210
Grid Snapping, 85
Grid Tool, 222
Grids Tab, 307
Group Object, 24
Group Tool, 198
GUI Windows, 278

H

Height Mapping, 105
Hide Geometry, 72
Hole Tool, 186
Hollow, 66
Hosts, 143
HPB Rotations, 66
Hyperboloid Object, 24

I

I.K. Tool, 264
ICO Output, 121
Icons, 349
If VSL Object, 109
Image Flare, 133

Image Flare Effect, 133
Image Gradient, 133
Image Object, 86
Image Processing, 104
Image Viewer, 346
Images Tab, 305
Impact Curve, 68
Impact Duration, 68
Impact Level, 69
In/Out Channels of VSL Object, 92
Indexing of rendered files, 7
Inertia, 68
Info Window, 295
Init UV Tool (SDS), 256
Initialize Ray, 101
Input Parameter, 288
Input Plane, 323
Input Plane Object, 25
Input Plane Tool, 222
Instance Object, 25
Instance Tool, 198
Integer, 170
Intensity of Light, 60
Interpolation of Bumps, 105
Interpolator Object, 25
Intersect Tool, 185
Intersection, 29
Inverse Kinematics Object, 75
Inverse Kinematics Tool, 190
Inversed Kinematics of Curve, 241
Invert Curve Tool, 241
Invert NURBS Mesh, 238
Invert Trim Side Tool, 243
IP by 3P, 210
IP by N, 210
Iso Value, 31

J

Join Faces Tool, 256
JPG Output, 122

K

Keyframe, 2

L

Language, 348
Lattice Mapping, 67, 212
Lattice Mapping Rotate Tool, 213
Lattice Mapping Tool, 212
Lattice Mapping/Rotate Tool, 216
Lattice Object, 76
Lattice Space, 67
Lattice Tool, 191
Leaf Object, 26
Leaf Tool, 244
Lens Flare Effect, 124

Level Material, 93
Level Object, 29
Level VSL Object, 109
Light Glow Effect, 134
Light Properties, 97
Light Source, 60
Light Source Tool, 216
Line Effect, 134
Line Tool, 175
Linear Interpolation of Channels, 112
Linear VSL Object, 109
Link Object, 29
Link Tool, 198
Load Window, 281
Look At Tool, 191
Loop Tool, 256

M

Macro Button, 278
Macro Slider, 278
Macros Tab, 311
Magnetism, 70
Map IK Tool, 207
Map Tool, 213
Map VSL Object, 110
Map2Obj, 220
Map2Obj Tool, 220
Mapped Shadows, 61
Mapping Object, 55
Mapping Tool, 218
Mass, 68
Material Browse Window, 295
Material Initialization, 95
Material Map, 55
Material Map Object, 55
Material Mapping Object, 55
Material Mapping Tool, 218
Material Preview, 87
Material Types, 86
Materials Tab, 302
Matrix, 156
Measure Curve Tool, 241
Measure Object, 30
Measure Tool, 208
Memory Usage, 144
Merge Objects Tool, 257
Mesh Map, 58
Mesh Object, 35
Metaball, 31
Metaball Tool, 198, 198
Metrics, 297
MipMapping, 115
Mirror Tool, 264
Moon, 52
Morphing Object, 32
Morphing Tool, 198

Move 1D, 200
Move 2D, 201
Move 3D, 201
Move Constructor, 76
Move Tool, 264
Multi Map Tool, 213
Multi Reflection, 141

N

Native File Format, 122
Network Rendering Daemon, 338
New Point to Curve Tool, 241
Noise Choreography, 3
Noise Tool, 265
Noise VSL Object, 110
Normalize Channel, 91
Normalize Object Space, 223
Numeric Window, 295
NURBS Circle Tool, 226
NURBS Cone Tool, 227
NURBS Cube Tool, 228
NURBS Curve, 32
NURBS Curve Tool, 228
NURBS Cut Cone Tool, 229
NURBS Cut Pyramid Tool, 229
NURBS Cylinder Tool, 230
NURBS Disk Tool, 230
NURBS Helix Tool, 232
NURBS Mesh, 35
NURBS Pyramid Tool, 232
NURBS Rectangle Tool, 233
NURBS Rotate Tool, 233
NURBS Sphere Tool, 234
NURBS Surface, 35
NURBS Sweep Tool, 234

O

Object Browse Window, 296
Object Color, 63
Object Knife Tool, 258
Object Properties Window, 298
Object Scripts, 69
Object Space, 66
Octaves of Noise, 110
OpenGL Tool, 210
Operation VSL Object, 111
Options Window, 296
Orthogonal Particles, 223
Outline Rendering, 144
Output Channel Bindings, 120
Output Object, 119
Output Objects Tab, 306

P

Packer, 208, 278
Packer Object, 76

Packer Tool, 208
paint with Selected Tool, 177
Parallel / Perspective Tool, 209
Parallel Curve Tool, 241
Parallel Map, 58
Parallel Map Tool, 220
Parallel Mapping Tool, 220
Parametrization, 33
Parametrization Tool (Curve), 241
Particle Disks, 125
Particle Effect, 135
Particle Lines, 126
Particle Tool, 199
Particles, 36
Paste Color on Objects, 176
Paste Colors Tool, 176
Paste Tool, 214
Path Level Constructor, 78
Path Level Object, 40
Path Tool, 215
Periodic, 146
Perspective Tool, 209
Phase, 146
Physical Properties, 68
Picker Tool, 176
Ping Pong, 106
Placer, 277
Plant, 46
Plant Seed, 46
Plant Tool, 244
PlantRoot, 10
PNG Output, 122
Point Light, 61
Point Light Tool, 217
Point Translator, 79
Point Translator Tool, 193
Polygon, 40
Polygon Painting Tool, 176
Polyhedron, 41
Polymid, 41
Pose, 2
Position Mark Object, 41
Post Image, 135
Post Image Effects Tab, 308
Post Particle Effects, 124
Post Particle Effects Tab, 309
Post Particles, 103
Post Process View Tool, 210
Post Processing, 142
PPM Output, 122
Priority, 55
Procedure, 92
Procedure VSL Object, 112
Projection, 15
Property Window, 298
Pyramid, 41

Pyramid Map, 58
Pyramid Map Tool, 220
Pyramid Mapping Tool, 220

Q

Quad View Tool, 212

R

Radial Deformator, 79
Random VSL Object, 113
Randomize Plant Tool, 244
Ray Direction, 96
Ray Hit, 102
Ray Termination, 103
Ray Tracing, 138
Raytrace VSL Object, 113
Realsoft Scene Viewer, 350
Rebound Energy, 68
Reconnect Tool, 216
Rectangle, 42
Rectangle Tool, 180
Rectangular Trim Curve Tool, 233
Recursion Depth, 139
Reduce Curve Degree, 242
ReduceU Tool, 238
ReduceV Tool, 238
Reflectivity of Surface, 97
Remove Tool, 185
Render Box Tool, 210
Render Settings, 138
Render Settings Tab, 304
Render View Tool, 210
Rendering Daemon, 342
Rendering Diagram, 104
Rendering Engine, 138
RendW, 342
Repeating Texture Sequence, 106
Replace Instances, 223
Replicator, 208
Replicator Tool, 208
Reset Object Handles, 224
Reset Plant Tool, 245
Reset View Tool, 209
Resolution of file rendering, 7
Rigidity, 68
Roof Line Tool, 187
Roof Object, 43
Roof Tool, 187
Room Object, 45
Room Tool, 187
Root Tool, 244
Rotate 1D, 202
Rotate 2D, 203
Rotate 3D, 203
Rotate Constructor, 80
Rotate Curve Tool, 242

Rotate Tool, 265
Rotation, 66
Rotation Tool, 265
Rotator Object, 42
Rotator Tool, 200
Round Curve Tool, 242
Row Column, 277
RsView, 350
RT-Invisible Tool, 224

S

Safe Region, 16
Safety Area, 142
Saturation Effect, 137
Scale, 66
Scale 1D, 204
Scale Tool, 265
Scanline, 103
Scanline Rendering, 142
Scope, 55
Script Choreography, 3
Scripting Window, 299
SDS, 48
SDS Add Tool, 250
SDS Bevel Tool, 251
SDS Circle Tool, 245
SDS Cone Tool, 246
SDS Cube Tool, 246
SDS Cut Cone Tool, 247
SDS Cut Pyramid Tool, 248
SDS Cylinder Tool, 248
SDS Delete Tool, 252
SDS Extrude Tool, 254
SDS Hide/Show Tool, 256
SDS Invert Tool, 256
SDS Merge Tool, 257
SDS Pyramid Tool, 249
SDS Rectangle Tool, 249
SDS Smooth Tool, 258
SDS Sphere Tool, 250
SDS Tri/Quad Tool, 260
SDS Weld Tool, 262
Secondary Ray, 102
Select Tool, 224
Select Window, 299
Selection Painter, 224
Selective Visibility, 73
Self Illumination Wizard, 118
SelPainterTool, 224
Separate Tool, 258
Separator, 280
Set Start of Mesh, 239
Shaded OpenGL Tool, 210
Shaded Quality, 73
Shader, 94
Shader Tree, 88

Shader VSL Object, 114
Shadow Map VSL Object, 114
Shadows, 61
Sharpen Curve Point Tool, 242
Sharpen Mesh Tool, 239
Sharpness of Specularity, 115
ShowImg Utility, 346
Side View Tool, 209
Simulation Properties, 70
Size Tool, 265
Skeleton, 80
Skeleton Tool, 193
Skew, 67
Skew Tool, 266
Smoothen/Flatten Tool, 259
Snap Tool, 225
Soft Selections, 225, 225
Soften Selection, 225
Solid Tool, 185
Special Light, 62
Special Light Tool, 217
Specular Color Wizard, 118
Specular VSL Object, 115
Sphere, 20
Sphere Map, 59
Sphere Map Tool, 221
Sphere Mapping Tool, 221
Spin, 68
Spin Tool, 263
Split Edge Tool, 259
Splitter, 281
Spot Light, 61
Spot Light Tool, 217
Start Tool (Curve), 242
Startopts, 348
Startup options, 146
Status Window, 313
Steel Wizard, 118
Stochastic Antialiasing, 140
Streaks, 125
Sub Flares, 125
Sub Wall Tool, 188
Subdivide Tool, 259
Subdivision Surface, 48
Sun, 52
SunMoon, 52
Surface Area, 30
Surface Filtering, 98
Surface Finishing, 100
Surface Friction, 68
Surface Geometry, 95
Surface Illumination, 99
Surface Properties, 96
Swap Curve Tool, 242
Swap Footprint Tool, 207
Swap Mesh Direction, 239

Swap UV of a Mesh, 239
Sweep Curve Tool, 243

T

Tab, 280
Tags, 71
Targa Output, 123
Texture Animation, 105
Texture Bindings, 116
Texture Bleeding, 105
Texture Indexing, 106
Texture Map Wizard, 119
Texture Mapping, 115
Texture Quality, 73
Texture Tiling, 105
Texture VSL Object, 115
Tiled Textures, 105
Time Curve, 146
Time Lines, 286
Time System, 297
To Face Material Tool, 262
To File Instance, 225
To SDS Tool, 260
To3D Tool, 243
ToBezier Tool (Curve), 243
ToCurves Tool, 234
ToNurbs Tool, 235
Tool Bar, 313
Tool Control Bar, 314
Tools, 174
Top View Tool, 209
Trace Distance, 114
Track Tool, 210
Transform to Material Space, 91
Transformation Tool, 263
Translation, 66
Transparency of Surface, 97
Transparency Wizard, 119
Triangles to Quads Tool, 261
Trim Curve, 53
Trim Curve Tool, 235
Trim On/Off, 243
Trim Tool, 235
Trim View Window, 314
Trim with a Curve Tool, 243
Triset Add Face, 267
Triset Auto Smooth, 267
Triset Delete Tool, 267
Triset Flat Tool, 267
Triset Flip Tool, 268
Triset Merge Tool, 268
Triset Object, 53
Triset Select Mode, 268
Triset Smooth Tool, 268
Triset Tools, 267
Tube Object, 54

Tunnel Tool, 261
Turbulence, 21

U

Undersampling, 140
Unified Circle Tool, 268
Unified Curve Tool, 272
Unified Rectangle Tool, 274
Unified Sphere Tool, 276
Unified Unified Creation Tools Tool, 268
Union Tool, 185
Unmap Tool, 216
Unwrap Tool, 261
Unwrap UV Tool, 261
User Defined Channel, 1
User Defined Color Channel, 1
User Defined Float Channel, 1
User Defined Vector Channel, 1
User Interface, 277
UV Editor, 314
UV Image Tool, 177
UV Property Window, 316
UV Set, 59
UV Set Connect Tool, 262
UV Set Coordinates From View, 262
UV Set Tool, 261
UVImage Painting Tool, 177
UVImage Tool, 177

V

V3 Material, 93
Variable VSL Object, 116
Variate Plant Tool, 245
Vector, 160
Vectorize Tool, 235
Velocity, 68
Velocity Tool, 263
Vertex Colors, 72
Vertex Painting Tool, 179
VF-Invisible Tool, 226
View Camera, 323
View Control Bar, 324
View Control Tools, 208
View Model, 145
View Models Tab, 312
View Popup Menu, 316
View Property Window, 327
View Window, 316
Visibility Options, 64
Volume Filtering, 98
Volume Illumination, 100
Volume Measure, 30
Volume Properties, 97
Volume Selector Tool, 226
Volume Shading, 101
Volumetric Light, 60

VSL antialiasing, 91
VSL Assignment Operation, 90
VSL Attributes, 89
VSL Effect, 138
VSL Input Channels, 92
VSL Material, 87
VSL Material Interface, 89
VSL Objects, 105
VSL Output Channels, 92
VSL Procedure Library, 92
VSL Samples, 93
VSL Shader Tree, 88
VSL Shaders, 94
VSL Wizards, 88, 117

W

Walk Thru Tool, 211
Wall Object, 54
Wave VSL Object, 116
Weight Curves, 290
Weight Tool (Curve), 243
Weight Tool (NURBS Mesh), 239
Weights, 290
Weld Tool, 239
Weld Tool (NURBS), 236
Weld Trim Curve Tool, 244
Window, 281
Window Output, 124
Windows, 282
Wireframe Properties, 71
Wizards, 88

X

XSect Tool, 236

Z

Zoom In/Out Tool, 211