



**Viz|Artist
User Manual
Version 2.8**

Copyright

Copyright © 2005- 2006 Vizrt. All rights reserved.

No part of this software, documentation or publication may be reproduced transcribed, stored in a retrieval system, translated into any language, computer language, or transmitted in any form or by any means, electronic mechanical, magnetic, optical, chemical, photocopying, manual, or otherwise, without prior written permission from Vizrt.

Vizrt specifically retains title to all Vizrt software. This software is supplied under a license agreement and may only be installed, used or copied in accordance to that agreement. All 'Viz|' followed by a name or number such as, but not limited to Viz| Trio, Viz|Artist, Viz|Content Pilot, Viz|Ticker 3D, Viz|Map, Viz|Weather, Viz|Ibis, Viz|Arena, Viz|FX, Viz|Virtual Studio and Viz|Engine are trademarks of Vizrt. All other product names are trademarks or registered trademarks of their respective owners.

Disclaimer

The information in this guide is believed to be correct as of the date of publication. However, our policy is one of continual development so the information in this guide is subject to change without notice, and does not represent a commitment on the part of Vizrt.

Technical Support

For technical support, and the latest news of upgrades, documentation, and related products visit the Vizrt Web site at www.vizrt.com.

Manuals were produced by:

Seraph International Business Service Group Ltd.
Beijing, China

Table of Contents

New & updated elements for 2.8 pl. 1	1
New & updated elements for 2.7 pl. 4	1
New & updated elements for 2.7 pl. 3	1
Welcome as Viz Artist user	
Viz Artist	1
[virtual set] expansion module	2
Vizrt support	2
Software Licence Agreement	
License	2
Installing Application Software	
New installation	3
Installing the program files on Windows systems	3
Installing the program files on a SGI Irix-based system	3
Viz Artist data directory	4
Viz Artist licence.	4
Upgrading	4
Upgrading the Viz Artist program files	4
Plug-ins	4
Installing Viz Artist plug-ins	4
Main menu	
The main menu buttons	5
Viz Artist databases	
Scene database	6
Object database	7
Material database	8
Font database	9
Image database	9
Function database	10
Standard functions	10
<i>Group function</i>	11
<i>Alpha function</i>	11
<i>Key function</i>	11
Key signal on graphics boards that support shader technology.....	12
<i>Expert function</i>	12
<i>Mask function</i>	12
<i>Z-Sort</i>	12

<i>Extrusion function</i>	12
<i>Shadow function</i>	12
<i>Projector function</i>	13
<i>Clipper function</i>	13
Other function plug-ins	13
Function plug-ins for scenes.....	14

Administration- and configuration views

Import view	14
Archive view (export)	15
Image export	15
Configuration Section	16
Targa 3200	16
Archive Paths.....	16
Audio.....	16
CGP Panasonic	16
Camera Set-up Configuration.....	16
Communication	16
Clock formats	16
Data root directory	16
Default FTP URLs.....	16
Device Manager setups	16
File Compression.....	17
Font Codes	17
<i>IME Editor</i>	17
<i>Font Handling</i>	17
<i>Use thread</i>	18
<i>2 Byte Texture</i>	18
Import 3D options	18
Import Merge Files	19
Import File Name Convert	19
Import paths	19
Video Input	19
Open GI Visual (Irix only)	19
Output GUI (Irix only).....	19
Output Screen Layouts.....	19
Renderoptions	19
Scene default values.....	21
Sensor.....	21
Targa3200	21
Video Output	21

Scene building objects

Viz Artist user interface	21
Scene Tree	22
Display Options.....	23
Tree search	24
The Rendering Area	24

Database editors and viewers

Material editor	24
Image editor	25

Font viewer	26
Scene editors	
Container editor	26
Transformation editor	27
Geometry editor	28
Texture editor	28
Text editor	29
Alpha editor	29
Key editor	29
Expert editor	30
Mask editor	30
Shadow Caster editor	30
Shadow Receiver editor	31
Extrusion editor	31
Projector editor	32
Projector Receiver editor	32
Light editor	32
Camera editor	33
Scene views	33
Editing in the three extra scene views	33
Control Channels	33
Stage editor	33
Customizing the Stage Editor	35
Stage directors	36
<i>Stage director timeline</i>	36
<i>Object animation timeline</i>	36
<i>Animation Channel timeline</i>	36
<i>Keyframe Editor</i>	37
The Animation Channels	37
Scene setup menu	
Background menu	38
Video menu	38
Audio menu	38
Rendering menu	38
plug-in menu	38
Clipper menu	39
Virtual Set menu	39
Device plug-ins	
Playback devices real-time and post	39
Record devices	39
Device Manager	40

Device control	40
Post Rendering Screen	41

Mouse clicks & shortcuts

Use of mouse and keyboard	42
Changing values.....	42
Reset functions	42
Scrolling content pages of databases	42
Scrolling the stage timeline.....	42

Using Viz|Artist databases

The folder tree	43
Using the Scene database	43
Editing a scene	43
Adding a Scene	44
Archiving a Scene.....	44
<i>Viewing at the elements to be archived</i>	44
<i>Deselect elements for archiving</i>	44
<i>Saving the archive</i>	44
Deleting a Scene	44
Using the Object database	44
Navigate the object database.....	44
Special on imported objects	44
Use an object in a scene container.....	45
Save an object in the database	45
Using the Material database	45
Use a material in a scene container.....	45
Add a new material	45
Edit a material	45
<i>Ambient color</i>	45
<i>Diffuse color</i>	45
<i>Specular color</i>	45
<i>Emissive color</i>	46
<i>Base color</i>	46
<i>Shininess</i>	46
<i>Alpha</i>	46
<i>Color picker</i>	46
Using the Font database	46
Use a font in a scene container	46
Editing a font in the font editor.....	46
<i>Changing font style</i>	46
<i>Changing font kerning</i>	46
Using the image database	47
Using an imported image in a scene container.....	47
Using a non imported image	47
Exporting an image	47
Grabbing a video still	47
Combine a RGB and an Alpha image	47
Using the Function Database	47
Built in standard functions	47
Container plug-ins	47
Scene plug-ins	47
Shader plug-ins	48

Viz|Artist Export and Import

Exporting	48
Importing	48
Open the import view	48
Browsing the network or disk for files to import.....	48
Connecting and browsing a FTP-server for files to import.....	48
Importing the selected files	48
<i>Import supported font types</i>	48
<i>Import supported image formats</i>	48
<i>Import supported 3D formats</i>	49
<i>Import supported materials and texture vertices</i>	49
Common problems with importing	49

Using scene building objects

Using the scene tree	50
Building, navigating & editing the scene tree.....	50
<i>Unmark a container</i>	50
<i>Edit a container name</i>	50
<i>Move a container</i>	50
<i>Copy a container</i>	50
<i>Create multiple copies</i>	50
<i>Change an object</i>	51
<i>Drag the same object to multiple containers</i>	51
<i>Drag an object directly to the rendering window</i>	51
<i>Delete a container</i>	51
<i>Delete an object within a container</i>	51
<i>Search the scene tree</i>	51
Using the Container Editor	52
Invoke an editor for an object in a container	52
Hide/show container or objects within a container.....	52
Delete objects or functions from a container	52
Split or merge a container	52
Save a compound object.	52
Using the Transformation editor	52
General description.....	52
Position an object.....	52
Scale an object	52
Rotate an object.....	52
Set the geometrical centre-axis	53
Screen position set by pixels.....	53
<i>Align to pixel</i>	53
Screen size.....	53
Follow functions	53
Face Camera	54
The object properties box.....	54
Transferring properties from one container to another	54
Using the Geometry editor	54
Object references	54
Deactivating a geometry reference	54
Reactivating a reference.....	54
Shading	55
Face orientation	55
Using the Texture Editor	55
Editing the texture properties of an image	55

<i>Mapping</i>	55
<i>RGBA information</i>	56
<i>Quality</i>	56
<i>Blur and sharpen</i>	56
<i>Position of texture</i>	56
<i>Texture optimizing and compression</i>	56
<i>Use of two images in one container</i>	57
Using the Text Editor	57
Editing text properties	57
<i>IME button</i>	57
<i>The Font properties</i>	57
<i>The Kerning properties</i>	58
<i>Shadow properties</i>	58
<i>Clock</i>	59
<i>Text box</i>	60
<i>Show icons</i>	60
<i>Using the IME-based editor</i>	60
Working in the Rendering Area	60
Editing containers in the rendering window	60
<i>Select and Position a container</i>	60
<i>Rotate an container</i>	61
<i>Move a container in Z</i>	61
<i>Copy a container</i>	61
<i>Editing with the right mouse button menu</i>	61
Additional functions connected to the output window	61
<i>The left side button bar</i>	61
Using the side-views	64

Using the main function plugins

Using the Key function	65
Using the Expert Function	65
Using Mask Function	67
Mask and Source layers	67
Inverted mask	67
Using the Window Mask function	67
Using Extrusion function	67
Extrusion, general description	67
Extrusion Editor parameters	67
Using the Shadow function	67
Using the Shadow Caster Function	68
Using the Shadow receiver function	68
Using the Projector function	69
General description	69
Projector parameters	69
Use of the projector function	69
Using the Clipper function	69

Lights and cameras

Using Light Editor	70
Editing the Light settings in a scene	70
<i>Switch lights on and off</i>	70
<i>Different type of lights</i>	71

<i>Position and direction of lights</i>	71
<i>Editing techniques for setting position and direction</i>	71
<i>Track object with light source</i>	71
<i>Animate Lights</i>	71
Using the Camera Editor	71
Editing the Camera settings in a scene	71
<i>Selecting camera</i>	72
<i>Camera views</i>	72
<i>Side views</i>	72
<i>Position, Zoom, Pan, Tilt and Focus</i>	72
<i>Shortcuts for orienting the camera</i>	72
<i>Camera shortcuts to the six orthogonal views</i>	73
<i>Track an object with the camera</i>	73
<i>Receive tracking data from a real camera</i>	73
<i>Copy properties from one camera to another</i>	74
<i>Animate camera settings</i>	74

Built in objects

Recurring parameters	74
Tessellation	74
Use LOD (level of detail)	74
Polygon	75
General description	75
Maneuver on the drawing pad	75
Select existing elements for editing	75
Altering the shape of the polygon	75
Add new intersection points within spline	76
Close/open the spline	76
Delete an intersection point or a border section	76
Copy/cut and paste intersection points	76
Spline handle modes	76
Set or hide anchor	76
Split spline sections	77
Flip spline sections	77
Mirror spline sections	77
Reset spline handles to original position	77
Move, Scale and rotate	77
Use linear lines between points	77
Add a new polygon	77
Add a new polygon at the top of an existing	78
Import Adobe Illustrator™ polygons	78
Short keys overview	78
Alpha Map	78
General description	78
Alpha map parameters:	79
Arrow	79
General description	79
Arrow parameters	79
Circle	80
Circle parameters:	80
<i>Circle tips</i>	80
Cogwheel	80
Cogwheel parameters:	80
Cone	80

Cone parameters.....	80
Cube	81
Cube parameters.....	81
Cylinder	81
Cylinder parameters:	81
Dexter	81
Dexter General description	82
Control	82
<i>Velocity control</i>	82
Design	82
Dexter design parameters	83
Dexter text editor.....	83
Trigger	84
Stopper	85
Dexter Control buttons.....	85
Script programming.....	86
<i>Tokens</i>	86
<i>Built in parameters</i>	92
Syntax rules	101
<i>\$\$</i>	102
<i>\$_</i>	102
<i>\$<, \$></i>	102
<i>Dexter basic function</i>	103
<i>Execution order</i>	103
<i>Two line example:</i>	103
Newline logic	104
Displacement Map	105
General description.....	105
Displacement Map parameters	106
Fade Rectangle	106
General description.....	106
Fade Rectangle parameters	106
Globe	107
Globe general description	107
Globe Parameters.....	107
Graph	107
Graph general description.....	107
Graph parameters	108
Icosahedron	108
General description.....	108
Icosahedron parameters.....	108
Pointer	108
General description.....	108
Pointer parameters	108
Rectangle	109
Rectangle parameters	109
RfxSmoke	109
General description.....	109
RfxSmoke parameters.....	109
ImageFX	112
.....	112
General description.....	112

How does it work	112
<i>Image</i>	112
<i>Geometry</i>	113
<i>Trigger</i>	114
<i>Effect</i>	116
<i>Control</i>	118
Noggi	118
Noggi parameters.....	118
Nquad	119
.....	119
General description.....	119
Nquad parameters.....	119
SplineFX plug-ins	119
2Dribbon.....	119
<i>Workflow</i>	120
2DPatch	121
<i>Workflow</i>	122
General.....	122
<i>Warning!</i>	122
<i>Tips</i>	122
<i>Known limitations:</i>	123
Ring	123
Ring parameters	123
Roll	123
General description.....	123
Roll parameters	123
Soft Clip Draw Pixels	124
The Soft Clip plugins.....	124
Soft Clip Draw Pixels general description	124
Codec support	124
CPU and I/O bandwidth.....	124
Loading multiple scenes	125
Half-height and Draw-Pixels.....	125
Ring buffers.....	125
Soft Clip parameters	125
Sphere	126
Sphere parameters.....	126
Spring	127
General description.....	127
Spring parameters.....	127
Torus	127
Torus parameters.....	127
Wall	127
General description.....	127
Wall parameters.....	127
Wave	128
General description.....	128
Wave parameters	128
Edit by using the mouse	128

Container plug-in functions

Grid	128
-------------------	-----

General description.....	128
Parameters.....	129
Circle Arrange	129
Circle Arrange general description	129
Circle Arrange parameters and functions	129
Time Displace	130
Time displace general description	130
Time displace parameters.....	130
Coco	130
Coco general description	131
Coco Parameters	131
Colin	131
Colin general description	131
Colin parameters.....	131
Cora	132
Cora general description.....	132
Cora parameters	132
Corena	132
Corena general description	132
Corena parameters.....	132
VCF_Param	133
VCF_Param general description	133
Prerequisites.....	133
How to.....	133
2D Follow	133
2D Follow general description.....	134
2D Follow setup	134
2D Follow parameters	134
Bar and Bar Values	134
Bar and Bar Values general description	134
Bar plugin	134
Bar parameters.....	134
Bar Values plug-in	135
Bar Values parameters.....	135
Pie Values and Pie Slice	135
Pie Slice and Pie Values general description	135
Pie Slice	136
Pie Values	136
TfxAlpha	136
TfxAlpha general description	136
Parameters.....	136
TfxArrange	137
TfxArrange general description.....	137
Parameters.....	138
TfxExplode	138
TfxExplode general description	138
Parameters.....	138
TfxJitterAlpha	139
TfxJitterAlpha general description	140
Parameters.....	140
TfxJitterPos	141

TfxJitterPos general description	141
Parameters.....	141
TfxJitterScale	142
TfxJitterScale.....	142
Parameters.....	142
TfxRotate	143
TfxRotate general description.....	143
Parameters.....	143
TfxScale	144
TfxScale general description	144
Parameters.....	144
TfxWrite	145
TfxWrite general description	145
TfxWrite parameters	145
Advanced counter	147
Advanced Counter general description.....	147
Advanced Counter parameters	147
Auto rotate	147
General description.....	147
Parameters.....	147
Colorize	148
General description.....	148
Parameters.....	148
Counter	148
Counter general description	149
Counter parameters.....	149
LOD	149
LOD general description	149
How to set up the LOD plug-in	149
LOD plug-in parameters	150
The LODManager plug-in	150
LODManager plug-in parameters	150
Magnify	150
General description.....	151
Parameters and functions	151
Omo	151
Omo general description.....	151
Omo parameters	151
Parlament	151
General description.....	151
Parameters and functions	152
Rotation Order	152
General description.....	152
Slide Show	152
Slide Show general description.....	152
Parameters.....	152
Soft Clip (Texture)	153
The Soft Clip plugins.....	153
Soft Clip general description	153
Soft Clip parameters	153

Temo	154
General description.....	154
Parameters:	154
Text Link	154
Text Link general description	154
Text Link parameters.....	154
Image Link	154
Image link general description	154
Image link description.....	155
TextParam	155
General description.....	155
Parameters to be animated.....	155

Shader Function plug-ins

Effect folder	155
Frame Mask.....	155
Frame Mask parameters	155
Image Mask.....	155
Image Mask Parameters	156
Soft Mask.....	156
Soft Mask Parameters	156
Water	156
Filter folder	156
Blend Image.....	156
Blend Image parameters	156
Blur.....	157
Blur parameters	157
Color Correction	157
Radial Blur	157
Radial Blur parameters.....	157
Sepia.....	157
Sepia parameters	158
Sharpen	158
Sharpen parameters	158
Material folder	158
Anisotropic Light	158
Cell Shader.....	158
Cell Shader parameters.....	158
Gooch.....	158
Texture folder	159
Drop Shadow.....	159
Emboss.....	159
Emboss parameters	159

Animation

Create an Animation	159
Animation menu.....	159
Director editor	160
Basic animation steps	161
Editing an Animation	162
Editing animations in the Stage Editor.....	162
<i>Adding Directors</i>	162

<i>Deleting Directors</i>	162
<i>Run a single director</i>	162
<i>Adding keyframes</i>	162
<i>Deleting keyframes</i>	162
<i>Adding stops</i>	163
<i>Adding actions</i>	163
<i>Adding Tags</i>	163
<i>Action keyframes</i>	163
<i>Editing keyframes</i>	164
<i>Change a moving object's path between existing keyframes</i>	164
<i>Change speed</i>	164
Explanation of animation channel parameters	165
Swing	165
Loop.....	165
Interpolate Smooth	165
Interpolate Linear.....	165
First/Last Linear	165
Linear Speed	165
Manual Speed	165
Close Path.....	165
Axis Interpolation	165
Quaternion Interpolation	166
Explanation of Keyframe parameters	166
Parameters common to all keyframes.....	166
Special parameters on position keyframes	166
Special parameters on rotation keyframes	167
Special on "effect" keyframes.....	167

Viz|Artist device plug-ins

Using the Device Manager	167
Enable and set up a plug-in	168
Using the Device Control Unit	168
Add a device to the scene stage	169
Clip Station post	169
Clip Station Control	169
ASC VR300	169
Accom Axess	169
Tektronix profile	169
Sony Generic	169
The Post Rendering Screen	169
Post rendering general description	169
Post rendering screen parameters	170
Post Render options.....	170
Render to disk	170
Render to Disk device, general description.....	170
Render to disk parameters.....	170
Setting up a video codec	171

Using the scene setup menu

Scene setup Background page	172
Scene setup Video page	173

Video menu	173
Video menu with Targa video card installed.....	174
<i>Possible modes for the channels</i>	175
<i>Parameters for the live video and the clip channels</i>	175
<i>Creating Targa clip animation keyframes</i>	176
<i>Post render to Targa playable clips</i>	176
<i>Configuration parameters for the targa functionality</i>	176
Extra video menu on CGP hardware from Panasonic.....	178
<i>Add clips to the stage</i>	178
<i>Clip file format</i>	179
<i>Create clips for the CGP 500</i>	179
<i>Performance</i>	179
<i>Limitations</i>	179
Audio	179
Installation, configuration and setup	179
Supported audio file formats.....	180
Play back audio files	180
Audio parameters.....	181
<i>3D audio</i>	181
<i>Doppler effect</i>	182
<i>Important note</i>	182
<i>Known Issues</i>	182
Rendering	182
plug-in	184
Clipper	184
Virtual Set	184

On Air Mode

On Air Info	185
--------------------------	-----

External Control Commands

Setup	186
Configuring the render machine and the control machine.....	186
Controller program.	186
<i>Make a buffer with commands ans send them:</i>	186
<i>Request an icon</i>	186
<i>To send multiple commands</i>	186
<i>Send a single command</i>	186
Internal Command Syntax	187
Number	187
Location	187
<i>Location specific syntax rules</i>	187
Commands	188
Command arguments datatypes	188
Overview of frequently used commands	189
About the examples.....	189
Use of RENDERER and SCENE as main location.....	189
Open and close scenes.....	190
Other Scene related commands.....	190
Hierachy related commands.....	190
Access and start animations.....	190
Directors, Keyframes and timelines.....	190
Actions	191

Jump the animation timeline	191
Create an animation channel.....	191
<i>Method 1</i>	191
<i>Method 2</i>	192
Set container transformation values.....	192
Set/change an image/texture in a container.....	192
Set/change a material in a container.....	193
Functions	193
Fonts and text	193
Create containers and load objects	193
Light and Cameras	194
Images and snapshots	194
Archive scenes	195
Material database.....	195
Change material on a container.....	195
Bounding box related commands.....	195

List of shortcuts in Viz|Artist

Camera shortcuts	197
Spline object shortcuts	197
Mouse/key on all value fields	198
Text kerning	199
Reset buttons in editors	199
Object pools	199
In stage	199
Render window	199
Main menu	199
Tree	200

Introduction

User Manual for Viz|Artist version 2.8.

New & updated elements for 2.8 pl. 1

- Key your scene with a button click. New Auto Key function for graphics cards that support shader technology. See, "[Scene setup Background page](#)" on page 172
- Texture Compression, see "[Texture optimizing and compression](#)" on page 56
- Illustrator import, see "[Import supported image formats](#)" on page 48
- Config section, some additions, see "[Renderoptions](#)" on page 19 and "[Scene default values](#)" on page 21
- Texture compression, see "[Texture optimizing and compression](#)" on page 56
- Some new parameters in Expert function, see "[Using the Expert Function](#)" on page 65
- Window mask, see "[Using the Window Mask function](#)" on page 67
- Globe geometry plug-in, see "[Globe](#)" on page 107
- Nquad geometry plug-in, see "[Nquad](#)" on page 119
- Soft Clip plug-in, see "[Soft Clip Draw Pixels](#)" on page 124 and "[Soft Clip \(Texture\)](#)" on page 153
- Shader plugins, see the chapter "[Shader Function plug-ins](#)" on page 155

New & updated elements for 2.7 pl. 4

- Dexter plug-in section updated, see "[Dexter](#)" on page 81.
- Paragraph about video features on CGP systems from Panasonic has been added, see "[Extra video menu on CGP hardware from Panasonic](#)" on page 178.
- New Audio feature described here: "[Audio](#)" on page 179.

New & updated elements for 2.7 pl. 3

- Targa 3200 paragraph has been updated. See, "[Video menu with Targa video card installed](#)" on page 174.
- 2D Follow plug-in, see "[2D Follow](#)" starting on page 133.
- Font Handling, see "[Font Handling](#)" starting on page 17.
- Photoshop file can now be imported, see "[Import supported image formats](#)" starting on page 48.
- New features in the transformation editor, see "[Screen position set by pixels](#)" starting on page 53. and "[Screen size](#)" starting on page 53.
- "Split times" function in Clock, see "Update Text" in "[Clock](#)" starting on page 59.
- Spline FX plug-ins updated with new parameters, see "Editing Range" and "Editing Mode" on "[2Dribbon](#)" starting on page 119. and "[2DPatch](#)" starting on page 121.
- Action keyframes - "Dir" property explained, see "[Action keyframes](#)" starting on page 163.
- Render to disk, frame formats updated, see "[Render to disk parameters](#)" starting on page 170.

Welcome as Viz|Artist user

Viz|Artist

Viz|Artist is an advanced real-time motion graphics authoring system for the creation of stunning real-time graphics. Built with an elegant and easy to use drag and drop user interface and sophisticated 3D animation and modelling tools, Viz|Artist enables the digital Artist to produce complex and engaging

visual content for broadcast, virtual set, and visualization in less time and with greater creative freedom. Viz|Artist is a software solution allowing users to create and compose their broadcast graphics, template graphics, or virtual sets. It is an authoring tool for both 3D and 2D graphics- including text. You can import 2D or 3D elements created with other software packages or they can be created within Viz|Artist. Scenes can be saved for playback via Viz|Engine. Viz|Artist can output at any required resolution, and is available with nearly identical functionality and appearance on all current Silicon Graphics systems. Differences in functionality are due to differing hardware capabilities in the various SGI systems. As system performance increases so does the possible scene complexity

**[virtual set]
expansion
module**

The [virtual set] expansion module transforms Viz|Artist to the only scalable, open and complete integrated solution for virtual studio production in the market today. Built with an elegant and easy to use drag and drop user interface, the most comprehensive studio and production workflow and phenomenal graphics capabilities, Viz|Artist with [virtual set] delivers the highest quality visual image to the most challenging productions. The [virtual set] expansion module integrates Viz|Artist with the widest range of tracking solutions and external studio peripherals and offers the most flexible set of price/performance options in the industry. Viz|Artist with [virtual set] expansion module is a complete software solution allowing users to go through the entire process of a virtual studio production including set creation, pre-production, studio set-up, rehearsal, production and post-production.

Vizrt support

At Vizrt we are committed to continuous enhancement to our products. We encourage your suggestions and feedback regarding the product and this documentation. Our support E-mail addresses is support@Vizrt.com - You are also welcome to use the web-based support at www.vizrt.com. You also have telephone support available from your local Vizrt contact.

Software Licence Agreement

License

The Viz|Artist software is the property of Vizrt and is protected by copyright law. While Vizrt continues to own the Software, you will have certain rights to use the Software after your acceptance of this license. This license governs any releases, revisions, or enhancements to the Software that Vizrt may furnish to you. Except as may be modified by a Vizrt license, or license key (each a "License Module") which accompanies, precedes, or follows this license, your rights and obligations with respect to the use of this Software are as follows:

You may:

- Use one copy of the Software on a single computer. If a License Module accompanies, precedes, or follows this license, you may make that number of copies of the Software licensed to you by Vizrt as provided in your License Module. Your License Module shall constitute proof of your right to make such copies.
- Make one copy of the Software for archival purposes, or copy the Software onto the hard disk of your computer and retain the original for archival purposes.
- Use the Software on a network, provided that you have a licensed copy of the Software for each computer that can access the Software over that network.
- After written notice to Vizrt, transfer the Software on a permanent basis to another person or entity, provided that you retain no copies of the Software and the transferee agrees to the terms of this license.

You may not:

- Copy the printed documentation which accompanies the Software;
- Sublicense, rent or lease any portion of the Software; reverse engineer, decompile, disassemble, modify, translate, make any attempt to discover the source code of the Software, or create derivative works from the Software;
- Use a later version of the Software than is provided herewith unless you have purchased upgrade insurance or have otherwise separately acquired the right to use such later version;
- Use, if you received the software distributed on media containing multiple Vizrt products, any Vizrt software on the media for which you have not received a permission in a License Module.
- Use the Software in any manner not authorized by this license.

Installing Application Software

New installation **Installing the program files on Windows systems**

Viz|Artist program files should be installed locally, to ensure maximum performance

- Insert the Viz|Artist CD into your CD-ROM drive. If the installation does not start automatically, select RUN from the STARTMENU and enter the command E:\SETUP.EXE (where E: stands for the drive-letter of your CD-ROM, if your CD-ROM is assigned to another drive-letter, substitute E: with your CD-ROM drive-letter). Alternatively, you can open the windows-explorer, change to your CD-ROM, and double-click on the SETUP.EXE.
- The setup procedure guides you thorough the installation process. You can choose between 3 setup types named "Typical", "Compact" and "Custom", or you can install only the Viz plug-in sdk named "plug-in SDK". We recommend that you use the "Typical" setup-type.
- After SETUP has completed copying the files to your disc, it installs the device driver for the dongle (hardware key).
- Viz|Artist is now installed on your system and ready for use on condition that the hardware key and license is installed.

Installing the program files on a SGI Irix-based system

1. Login as root user. If you have downloaded a tar compressed file from our ftp site, continue at point 2. If you install from a CD, jump directly to point 5.
2. Create a directory called Viz under /tmp of your machine
3. Copy from the file named Vizirix25pl13.tar into that directory. (This file name means "Viz for irix version 2.5 patch level 13")
4. Untar (tar -xvf) the installation file Vizirix25pl6.tar. It then creates a dist installation directory.
5. Open the software manager (swmgr)
6. If you have unpacked the tar file as described above, type in the path /tmp/Viz/dist and click lookup. If you install from a CD type the path CDrom/dist and click lookup. In both cases a Viz installation window will open. Read the text and press the Execute button.
7. When the installation runs a shell will open and will ask you some questions. In principle you can normally accept the values that the installation suggests by pressing enter after each question. You can of course change the settings if you want to. It will ask you:
 - To enter a user name. Type in: "Vizuser".
 - Which path you want as the data directory default path. It suggests (/usr/people/Vizuser/data). If this is not suitable, define another data-path.

8. When you get a message saying "Installation complete..." close the shell and press customized installation.
9. Mark the product Viz 2.5 patch level 6 and press start
10. After the installation is done, exit the Software manager and logout
11. Login to the Vizuser account.
12. From the toolchest start Viz (not on-air). It will open a window asking for the license string.
13. Print the license file you got from Vizrt, and type in your specific license string, into the window. Note: Type in lowercase & couples (like: cc 1a 4b....etc.)
14. If the license string is correct Viz will start. If not type it again. If you have problems, also check the sysinfo of your SGI machine and make sure it matches the one specified in the license file. To renew a existing licence, open the licence window by pressing the "i" button to the right in the main menu and type inn the new license key.

Viz|Artist data directory

It is quite common to have the data directory on a shared network server. By default Viz|Artist is set up to work on a local data area. The data directory is by default "Viz installation direcorey"\data.(If you installed Viz|Artist in c:\program files\Vizrt\Viz, the data directory is c:\program files\Vizrt\Viz\data. To change this you must start Viz|Artist and enter a new data root directory in the Config section. See ["Data root directory" starting on page 16](#). It is also possible to start Viz|Artist in a config mode: add the option -c in the start-up line in the program start-up icon, "C:\Program Files\Vizrt\Viz\Viz.cmd" -c

Viz|Artist licence.

On windows systems Viz|Artist requires a dongle (hardlock), to operate correctly, so make sure that the hardlock dongle is properly connect to the parallel port of your computer!

If you start Viz for the first time, you may need a license for your system. To get a license, you need to contact Vizrt by phone: +47-55 90 84 20 or (preferred) e-mail to license@Vizrt.com. You need to submit the dongle serial-number and the hardware on which Viz should run on (e.g. SGI VisualPC 320; SGI VisualPC 540).

Upgrading Upgrading the Viz|Artist program files

Before you upgrade, make a file copy of the existing program files or make sure you have a recent backup on tape or equivalent. To install a upgrade, the method is just the same as with a first time installation. You place the upgrade/fix over the existing installation.

Plug-ins Installing Viz|Artist plug-ins

The plug-ins are located in the folder named "plug-in" in the Viz main folder. They have the file extension ".evp" on windows based systems and ".so" on SGI Irix systems. The installation of plug-ins can be made simply by copying the new plug-in files to the plug-in folder in your Viz|Artist installation. However most plugin are distributed in installers. Some plug-ins require a new license code to work. Contact Vizrt to obtain your license key!

Getting to know Viz|Artist

Main menu

The main menu buttons

After starting up Viz|Artist the Screen shows the contents of the Scene database. At the top of the screen at the left side the buttons allow you view the different databases:



- **Scene** opens the Scene database (Shortcut-F1). To find out more about the scene database, see paragraph ["Scene database" starting on page 6.](#)
- **Object** opens the Object database, which holds imported objects and "Built in" graphical objects and primitives (Shortcut-F2). To find out more about the Object database, see paragraph ["Object database" starting on page 7.](#)
- **Material** opens the Material database (Shortcut-F3). More details about the Material database, see paragraph ["Object database" starting on page 7.](#)
- **Font** opens the Font database (Shortcut-F4). To learn about the Font Database, see paragraph ["Font database" starting on page 9.](#)
- **Image** opens the Image database (Shortcut-F5). To learn about the Image Database, see paragraph ["Image database" starting on page 9.](#)

You may switch between the databases at any time. On the right side of the main menu you will find:



- **Import** opens the import view. Here you can import scenes and parts of scenes as images, materials, fonts and objects. More on import see, ["Import view" on page 14](#)
- **Config** opens the config section where you set different options for the program functionality. For details on configuration see, ["Configuration Section" on page 16](#)
- **Log** displays application error logs e.g. if an image import fails, or a device fails to connect. The log button changes its color to red when the error log receives a new entry.
- **Post** opens the post rendering view. This button will only show if the post rendering option is installed.
- **On Air** switches the application from a GUI modelling tool, to a rendering engine (Shortcut-F12). The application now waits for control commands on its communication port. The default port is 6100 but this can be changed in the config section.

After having opened or created a scene the menu-bar expands. The following items get added in the middle section of the menu bar:



- **Function** opens the function database (Shortcut-F6). To find out about the function database, see paragraph ["Function database" starting on page 10.](#)
- **Control** displays the Control Channel Window (Shortcut-F7). See ["Control Channels" starting on page 33.](#)
- **Tree** displays a larger view of the Scene Tree (Shortcut-F8). To read description of this view, see paragraph ["Scene Tree" starting on page 22.](#)

- **Views** displays three extra scene views (Shortcut-F9). See ["Scene views" starting on page 33](#).
- **Stage** displays the Stage Editor where you edit and control animations and automation (Shortcut-F10). To learn about the Stage Editor, see ["Stage editor" starting on page 33](#).

These are also present when a scene is loaded:



- **Container** displays the Container Editor. See ["Container editor" starting on page 26](#).
- **Camera** displays the Camera Editor. To learn about the camera editor, see ["Camera editor" starting on page 33](#).
- **Light** displays the Light Editor. To learn about the Light editor, see ["Light editor" starting on page 32](#).
- **Setup** displays the setup-submenus for the scene. See ["Scene setup menu" starting on page 38](#).

To the upper right you find the following buttons:



The black arrow allows you to minimize the application



The black X allows you to exit Viz|Artist.

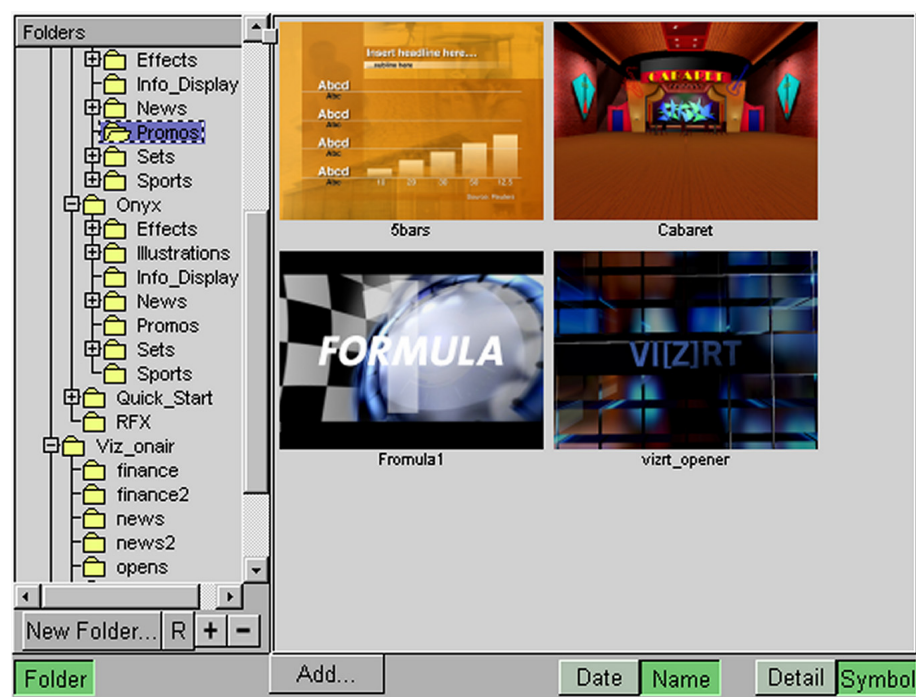


Opens the license information editor. When your existing licence expires, you can enter your new licence key here!

Viz|Artist databases

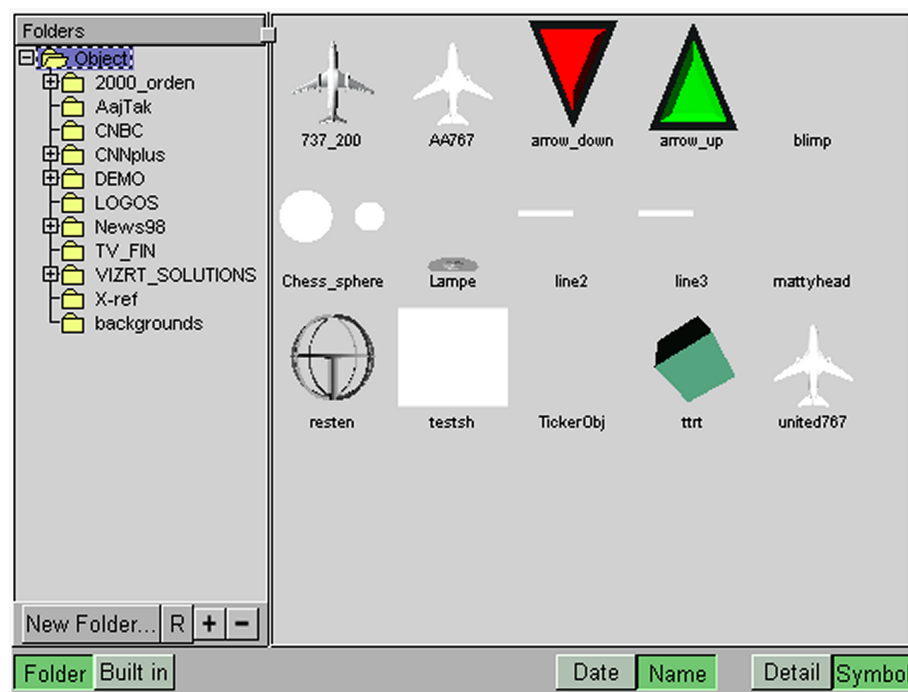
Scene database

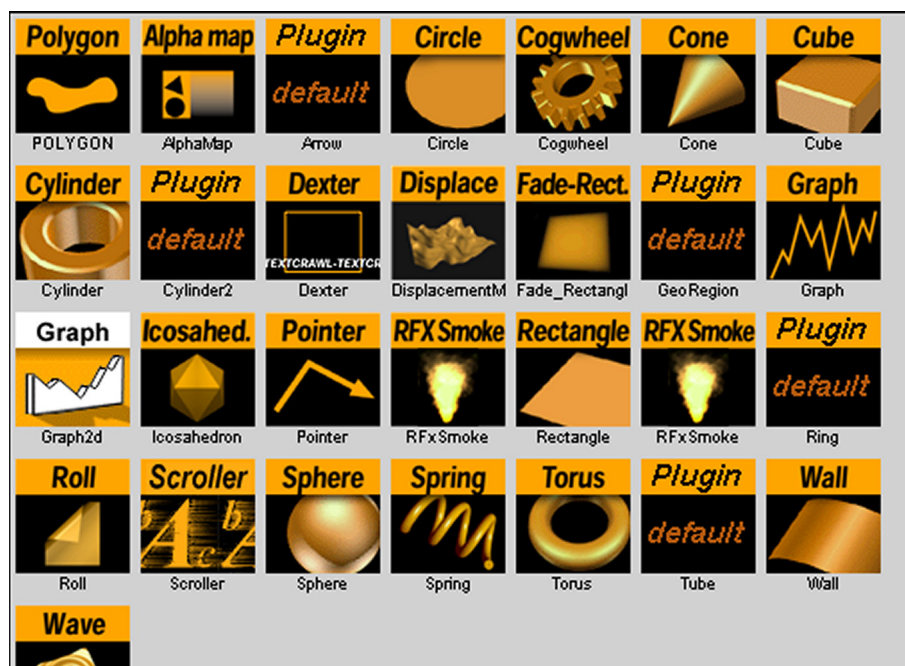
You open the scene database by clicking on the Scene-button in the main menu bar. This database holds all the scenes you have created so far. Each Scene is represented by an icon generated from the appearance of the scene when it was last saved. These are displayed to the right in the Scene Database. To the left is a folder tree which represents the sub folders of the scene-folder in the Viz|Artist data directory. Browse this to find the scene you want to open. With no scene opened you can access the archive view by clicking on the archive button on the lower right side of the screen. Details on how to archive a scene or scene elements see, ["Archiving a Scene" on page 44](#)



To learn more about the Scene database, see [“Using the Scene database” starting on page 43.](#)

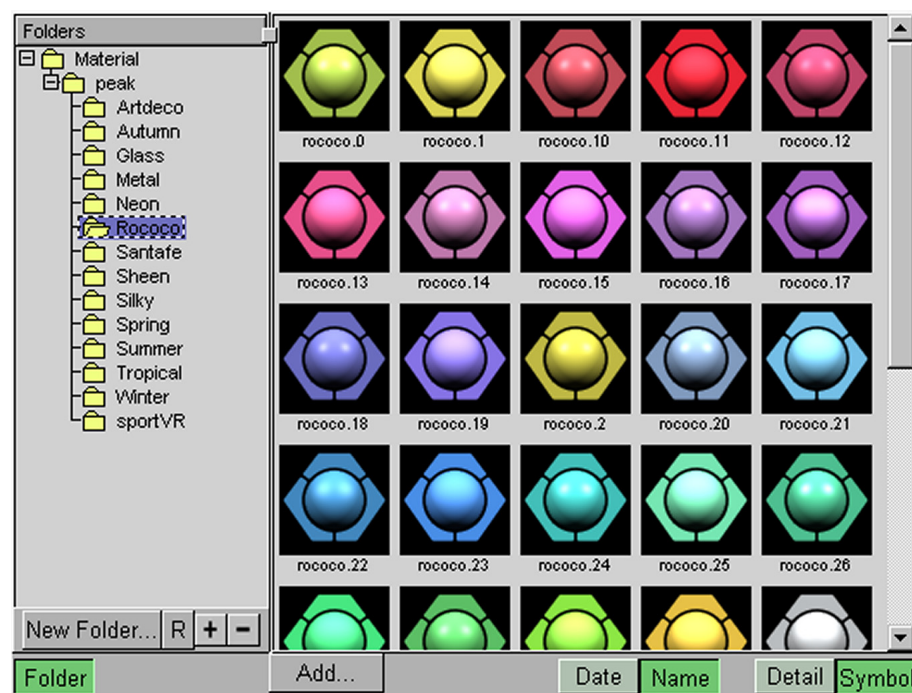
Object database





You open the object database by clicking on the Object-button in the main menu bar. When no scene is opened yet, this database holds only imported and compound graphical objects. When a scene is loaded it also gives you access to the Viz|Artist built in objects. To switch between "imported/compound objects" and "built in objects", click the buttons "Folder" or "built in". See more on the use of the object database in the paragraph ["Using the Object database"](#) starting on page 44.

Material database



This database holds materials with different color-, light- and shade effects. You can add new materials and edit existing ones. If you double click on an existing material or create a new one, the Material Editor opens and you can set the properties for that material. See: [“Using the Material database” starting on page 45.](#)

Font database This database contains the fonts you have installed. By double clicking on a font you open a font viewer, where you watch an test the font properties to check that they have been imported correctly. How to use the Font Database, see the paragraph [“Using the Font database” starting on page 46.](#)

Picture showing the font database

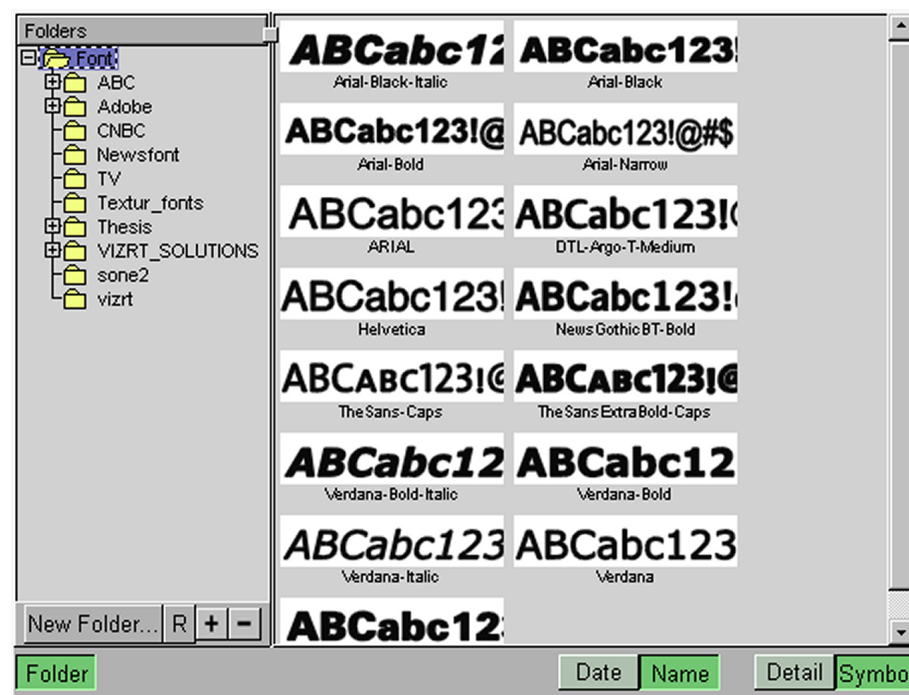


Image database The image database opens when you click on the image button in the main menu. It can be accessed in two modes, both with and without a scene loaded. The two modes differ on some elements. The “File” button is only visible when a scene is loaded and the “images combine” function is only available when no scene is loaded. The image database contains the following views and functions:

- Imported images. These are shown when you click on the “Folder” button. To the left a folder tree is visible. This folder structure represents the image folder in the Viz|Artist data directory and its sub folders. You can browse through the images by clicking on the different folders in the tree. You can add and delete new folders and files. By double clicking on an image, you invoke the image editor. See [“Image editor” starting on page 25.](#)
- Grabbing of video stills. This can be done in the window that opens when you click on the “Built in” button. This of course presupposes that the machine you are working on have a video card installed so you can receive video input. To set up and define the video input see, [“Video Input” on page 19](#)
- Access to images not located in the Viz|Artist data directory. To browse for files lying outside the programs own data-area, you can click on the “File”

button and a browser window will appear. (This button is only visible when a scene is loaded). Image files that the program recognizes can be added to a container. This feature is only meant for image testing purposes, and should not be used when you build permanent scene elements.

- Export of images. By dragging an image (in the folder view) to the Export icon at the bottom bar of the image database, you can export an image to disk in the following formats:
 - SGI
 - Jfif
 - Alias
 - Tiff
- Combine a RGB- and an Alpha image to a RGBA-image.

Picture showing
the image
database view



Function database

Standard functions

You access the function database by clicking the "Function" button in the main menu. The functions are displayed as icons. To apply a standard function in a container in the scene tree you drag it from the function database and onto the container. Below are the Viz|Artist standard functions listed. Most of them have its own editor where you set the parameters of the function. Links to paragraphs explaining the corresponding editors are listed under the functions.



Group function



The Group function is a container with no object that you can use to group elements in the scene tree. However all types of containers can in principle act as a “group” container. When one or more containers are placed as sub elements of another container or a group container, a parent-child hierarchy is established. A child container will inherit the transformation properties of its parent or parents. If you for instance have an object that is animated to move in a straight line from the left to the right side of the screen, and you create a rotation animation on its parent container, then the object will both move and rotate. If you thereafter place the child and the parent under different stage containers, you will be able to play of the different animation channels separately. This is a typical way to create a set of animation variations in a scene. For more details on the animation stage, see “[Stage editor](#)” on page 33

Alpha function



Alpha. This function adds an alpha channel to the container. The alpha channel defines the degree of transparency for the container and its subcontainers. Editor: see “[Alpha editor](#)” on page 29

Key function



Key, adds a Key signal to a container. Editing the key properties see, “[Key editor](#)” on page 29

Key signal on graphics boards that support shader technology

From version 2.8 pl.1 Viz|Artist can create a key signal by using a PreShaped/Key rendering. This basically means that you can create a key signal that is always correct by just clicking a button. Read about **Auto Key** in ["Scene setup Background page" on page 172](#).

Expert function



Expert, sets some special properties and adds some advanced functions to a container:

- Draw Mode, Normal, Wireframe, Outline
- Back face
- Two-sided lightning
- Z-buffer draw
- Line Anti aliasing
- Mirror
- Render mode

Editor, see ["Expert editor" on page 30](#)

Mask function



The mask function consists of two icons:

- Mask Source, adds mask source properties to a container.
- Mask Target, adds mask target properties to a container.

Editor, see ["Mask editor" on page 30](#)

Z-Sort



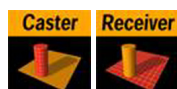
This function disables the automatic z-sort for a container. How the elements in the container get sorted in the Z-buffer is then dictated by the objects places in the container hierarchy. The containers at the root of the tree are drawn first and the containers at the bottom last.

Extrusion function



You use this function to create an extrusion on a 2-D primitive or a font. Editor see ["Extrusion editor" on page 31](#)

Shadow function



The Shadow function is displayed as two function icons:

- Shadow Caster, sets the properties for the shadow caster container. Editor for the Shadow Caster, see ["Shadow Caster editor" on page 30](#)
- Shadow Receiver, sets the properties for the shadow receiver container. Objects that can function as a shadow receiver must be built in 2-D primi-

tives. An imported 2-D will not work. Editor for the Shadow receiver, see ["Shadow Receiver editor" on page 31.](#)

Projector function



The Projector function allows you to set up one or more containers to project a planar display of its texture onto one or more containers. The function is displayed as two function icons:

- The Projector, sets the properties for the container that is to project its texture onto the projector target, see ["Projector editor" on page 32.](#)
- The Projector target function, sets the properties for the container that is to receive the texture projection from one or more projector containers. Editor, see ["Projector Receiver editor" on page 32](#)

Clipper function



The clipper function is an alternative to the mask function. Objects with the clipper function attached that are placed behind the clipper plane will be masked/clipped. If an object is only partially behind a clipper plane, only the part of the object that are behind the plane will be clipped out, since the clipping is done in through 3D space. The function uses the OpenGL clipping planes. On use see, ["Using the Clipper function" on page 69](#)

Other function plug-ins



You access these functions by clicking the Container button within the function database. The folder list to the left contains different plug-ins organized by type. Click on each folder to display the contents. Which plug-ins you see here depends on which plug-ins you have installed. How to install plug-ins, see ["Plug-ins" on page 4.](#)

Overview of the container plug-ins in a Viz|Artist basic installation, see [“Container plug-in functions” starting on page 128](#).

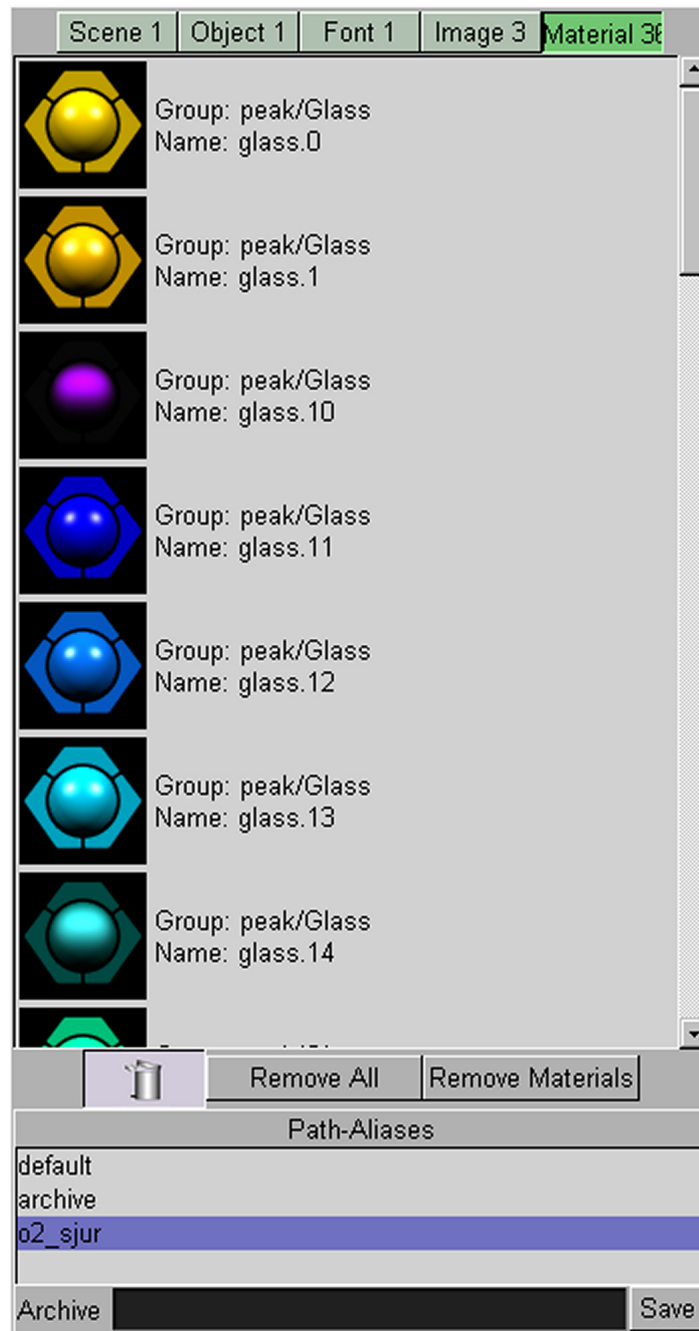
Function plug-ins for scenes

The Scene plug-ins are functions that affect the whole scene, not a single container with its sub-containers, as the case is with container plug-ins. The drop-zone for a scene plug-in is located in the setup-menu under the sub section named “plug-in” (setup button in main menu). More about scene plug-ins, see [“Scene plug-ins” on page 47](#)

Administration- and configuration views

Import view

To open the import section in Viz|Artist, click the import button in the main menu bar. Viz|Artist have the ability to import images, fonts, geometry and full archives (see scene database). Import can be made both from the network file system and from a FTP-server. When importing from the network, file listings can be sorted by date or by name and you can select which parts of the archive you want to import. When using the FTP-import, only complete archives can be imported. You can switch source directories by selecting a path alias in the lower left. The directory structure is shown in the left part, the objects are shown in the right part. At the top left you can choose which object type Viz|Artist searches the disk for. More on import, see [“Viz|Artist Export and Import” starting on page 48](#).

**Archive view
(export)**

You can archive a scene and all the objects within. This is done in the Scene database view with no scene loaded. The archive view opens when you click on the Archive button located in the lower right corner of the screen. More details on archiving, see paragraph ["Archiving a Scene" starting on page 44.](#)

Image export

In the image database you can export an image to disk in the following formats: .sgi, .jif, .alias, .tiff and .tga (pl.13). More on export, see ["Exporting an image" starting on page 47.](#)

**Configuration
Section****Targa 3200**

For targa parameters see targa paragraph, ["Configuration parameters for the targa functionality" on page 176](#)

Archive Paths

Here you can set five different aliases for use with the archive function (see ["Archiving a Scene" on page 44](#)). You type in a name that you want to use and apply a path. For selecting the path you can click the "Browse" button or you can enter it manually.

Audio

Here you can switch off the audio for Viz, select the default path for audio files and select the audio device. These are the audio capable devices installed in your system (usually only one). If you have a targa board installed, you can select Targa as audio device and then the audio output will be embedded in the targa video out stream.

CGP Panasonic

On Panasonic hardware you must set the Clip data directory here.

Camera Set-up Configuration

This section is only applicable if you have purchased the [virtual set] option. Here you define the configuration of the connected real cameras. The necessary values are normally defined and entered at installation time by a qualified systems engineer and therefore not intended to be changed by the user.

Communication

Any external control software (e.g. Viz|Content Pilot) uses TCP/IP network connections to send its commands to the Viz|Artist rendering engine. Viz|Artist expects its command at a certain port which has to be defined here. For any single pipe (one display) system, the default value (6100) is fine and should normally not be changed.

Clock formats

Here you can define up to 20 different formats of the digital clock object which is found in the Text object, see ["Clock" on page 59](#).

Data root directory

Here you enter the path to the Viz|Artist data root directory. You can browse to it by pressing the browse button. It is also possible to enter an UNC path (Uniform Naming Convention, the basic format is: "\\servername\share-name). By using this method you avoid problems with users not having the correct drive mapping.

Default FTP URLs

Here you can define up to 5 different FTP aliases. Give the alias a name, enter user-id, password and the URL to the site. Save and restart the program to apply the settings.

Device Manager setups

This sections helps you setting up devices that Viz|Artist interfaces with. see paragraph ["Using the Device Manager" starting on page 167](#).

File Compression

To save disk space and to speed up scene loading, Viz|Artist stores its data in a compressed form (similar to tools like "compress" or "gzip"). On remote (nfs) mounted file systems this also reduces network load dramatically during a scene load. In this screen you can switch on and off this compression for certain object types. By default compression is enabled for all object types.

Font Codes

This section allows you to choose the font encoding for Viz|Artist. You can choose between:

- Default
- Japanese Industry Standard Code
- Shifted Japanese Industry Standard Code
- Extended Unix Code
- Unicode
- UTF-8

If you have selected font encoding different than "default" Viz assumes you are using 2 byte characters, which can not be entered using the regular editor. A 2 byte character is a character that can't be represented by using one byte (max value 255). Viz will then by default try to open the IME editor when you drop-in a new font. Old fonts (imported before version 2.6) will not default to that since they don't always work with the IME.

IME Editor

When writing text in Viz|Artist you can choose to use an IME editor (a button in the text editor enables the IME input mode). Latin based languages (English, German, French, Spanish, etc.) are represented by the combination of a limited set of characters. Because this set is relatively small, most languages have a one-to-one correspondence of a single character in the set to a given key on a keyboard. When it comes to East Asian (EA) languages (Chinese, Japanese, Korean) the number of characters to represent the language can be in the tens of thousands, which makes the using a one-to-one character to keyboard key model next to impossible. To allow for users to input EA characters, several Input Methods (IM) have been devised to create Input Method Editors (IME).

An IME is a program running in windows, that allows computer users to enter complex characters and symbols, such as Japanese characters, using a standard keyboard. In order to begin entering for instance Japanese characters in an application running on Windows 2000/XP, you need to activate the IME by selecting it from the list of input languages. When you activate the IME, the floating Language bar in windows changes to the Japanese IME toolbar.

You can set the IME editor in Viz|Artist to support:

- Unicode - is normally the one you need but only if you are using 2 byte characters. A character which can't be represented by using one byte (max value 255) is called a 2 byte character.
- Complex Script - The only reason to use "Complex script" is if you are using a left to right language or a complex script language and you want to save a scene on windows which later will be loaded on Irix. If you don't have any Irix machines you should not use the complex scrip option in the IME editor.

Font Handling

Viz can handle the text layout in 2 ways. "Font File" is based on information which is stored in the font file simply by applying offsets (kerning) after drawing each character. But for some languages that is not good enough: Complex script languages, where one character can be composed from a few glyphs, can't be handled based on kerning only, and right to left languages are dis-

played wrongly when writing numbers. The solution was to use a windows library to handle that: "Complex Script"

So what should I select?

If you are using a left to right language which is not complex script (this is mostly the case), select "font file", If you are using right to left or complex script select "complex script".

If you choose Complex Script as you setting, Viz|Artist will activate an improved support for complex script fonts on a windows platform. The benefits are:

- Correct handling of numbers in right to left languages
- Mixing left to right and right to left in the save text (but only if the font itself supports both)
- Text box support for text written from right to left

In order for this to work the font **MUST** be installed in windows and the Viz font file **MUST** be "glyph indexed" (true for fonts imported into Viz with version 2.6 and higher). If things are not working as they should, try to reimport the font from windows/fonts.

Use thread

Creating textures for multi byte fonts takes performance from the system. During this process interruptions in the rendering process can occur. To avoid this one can enable the Use thread option. The texture generation will then be done in a separate thread in the program, thus not inflicting the rendering process. However, with this option on, fonts might appear a while after the scene is loaded, so it can only be used in situations where the fonts are created "off screen" such as with a ticker system.

2 Byte Texture

Here you can choose whether to optimize the texture generation for multibyte fonts, either for speed or quality.

Import 3D options

Here you can define certain parameters influencing the import of 3D objects (separately for all available formats). For each format you have three controls available:

- **Unify:** if activated, all vertices are recalculated during import in a way, that the object centre is moved to the origin (0/0/0) and the size of the object is 100cm in its largest extend. If deactivated, all vertices maintain their values as defined in the original file. Therefore it may happen that an object is invisible after putting it to a scene because the object is translated a lot from the origin or is scaled up or down very much. You may need to deactivate the unification to be able to recombine several separately imported objects which have to keep their size and relative position.
- **Face orientation:** Polygonal 3D models often do not have a consistent face orientation, but for performance reasons the Viz|Artist renderer expects that all faces of an object point to the same direction. If you activate this function, Viz|Artist tries to rearrange the orientation of the object faces during import. This may not always be correct, but in most cases the result is quite ok.
- **Shading Angle:** If you have a 3D object with no normal vector information, Viz|Artist automatically recalculates the normals from the geometry to make lightning possible. This recalculation is influenced by this angle which acts as a threshold between sharp and soft edges. A value of 60 for example means that an edge between two faces is considered to be a soft edge for angles below 60 and a sharp edge above this level. 60 is also the default.

Import Merge Files

Viz|Artist can automatically import and combine image and alpha information stored in different files. Here you can specify this naming convention and Viz|Artist at import time will read both files and construct one RGBA image.

Import File Name Convert

Here you can set five different aliases for use with the import path function. You type in a name that you want to use and apply a path. For selecting the path you can click the 'Browse' button.

Import paths

Here you can create path aliases to locations from where you often import objects and other scene elements.

Video Input

If applicable you can here define up to four video input formats. How many video channels you have depend on your hardware configuration.

Open GI Visual (Irix only)

This editor enables you to view and change the hardware configuration of bitplanes. The display of hardware bitplanes is done with the following window. Be careful with changes you make to this editor unless you know exactly how it works. You can see here if, for example, the hardware can do screen anti aliasing by multisampling or how many Z-buffer bitplanes are available.

Output GUI (Irix only)

This section is only applicable if you have a multichannel capable computer (like the SGI ONYX2). It basically defines on which channel the Viz|Artist authoring environment will appear after startup or if it should cover the whole screen (all channels).

Output Screen Layouts

Here you define the output format of the Viz|Artist rendering engine.

- Most known PAL, NTSC and HD formats are supported.
- USER DEFINED: With the user defined selection you can define your own format and frequency as well as the position of the output window in respect to the top left corner of the screen. After making a selection please make sure that the physical refresh rate of the graphics hardware and the video hardware is configured correspondingly. If user defined is selected, you also have the possibility, in the right part of the editor, to set up the viewing frustums in a special way to fit the requirements of multipipe systems (used for video walls or visionariums). The setup for these systems has to be done by experienced system engineers and therefore these parameters are not explained here in detail. For more information please contact your local support or Vizrt directly.
- When PAL, NTSC or HDTV are selected, you can set the change the aspect ratio of the format between "4:3" which is for a standard TV and "16:9" which is for wide screen TV.

Renderoptions

This section defines the capabilities of the render window. In the bottom line you can see if your current selection is valid for the hardware you are running on. You may find that not all capabilities can be selected at the same time on certain platforms

- **Line Antialiasing:** If the machine you are running on does not have full scene antialiasing, Viz uses antialiasing of lines to smooth edges on objects. This only works on 2D objects (rectangle, text ...). In some cases it has happened that Viz did not recognize the full screen antialiasing support on some graphics cards. Therefore this option was needed to switch line AA off.
- **Extended colorspace:** This influences how the conversion from RGB to YCbCr colorspace is done. Normally values from 0.255 in RGB space are mapped to 16..235 for Y (luminance) and 16..240 for CbCr (chrominance). In Extended Colorspace (which is only available on Voodoo boards) the mapping is 1:1, i.e. 0..255 in RGB are always mapped to 0..255 in YUV. This means that scenes will look different when you enable Extended Colorspace (so it's nothing you should do for existing scenes) but gives you the possibility to output illegal colors which might be needed in some rare cases (e.g outputting superblack).
- **Force sleep:** G-force graphics cards can use a lot CPU when a swap buffer is done. Viz renders into the back buffer and waits for the vertical retrace to swap buffers. This waiting can use a lot of CPU. If this happens the GUI can not redraw itself and is very slow. The force sleep measures the time to the next vertical retrace and calls the swap buffer command after a "sleep". The sleep pauses the renderer for a short time and does not use any CPU. This way the "non updating GUI" problem on single CPU machines with g-force cards is fixed. Viz can not do the sleep automatically, because "sometimes" the sleep sleeps too long (a windows problem) and Viz loses a frame.
- **Keep editing aspect:** Allows designers to maintain a correct graphics aspect while working on 16:9 scenes. It will squeeze the Y axis to compensate for the FOV squeeze of the X axis. This will not look correct on the video monitor of course. The mode can be set to be active in two cases:
 - Editor - When in editing mode
 - No video - When system has no video output.
- **Display list:** This is an open GL feature. If you render an object you can send the vertices every single frame to the card or define a display list and call the display list every frame. This is a much faster way of rendering geometry, but it only makes sense for "static" objects. An animated cylinder changes its vertices all the time so defining the display list does not make sense. There were some problems with display lists on some laptops and on some G-force cards, so you can force Viz not to use display lists at all.
- **Shaders** - Sets the Shader functionality active.
- **Internal Chromakey** - This option is relevant for Virtual Studio preview. If set, the camera signal is fed into viz and then the graphics are keyed over the camera using a shader. This can be used for a computer which is designated to be a preview machine in a studio so the cost of an external keyer can be saved. The quality of the key is not for broadcast but for preview only and that there is a performance cost. But for preview purposes this is acceptable.
- **Video Output window** - Having the Output window On Screen is default. However when the window is visible on screen, objects that are put over the output window will be part of the video signal. On single machine configurations where the control software is running on the same machine as the renderer, the Video output window should be set to Off Screen. You will then not see the preview, but the video signal is then not inflicted by the client gui. This feature is only supported on Nvidia Quadro graphics boards.
- **Full Scene AA** - If the settings for the graphics card is set to "Application controlled" the antialiasing settings here will be used.
- **Shaped Fill** - Enables a faster way of drawing key when auto key modes is selected for the scene. It is using a shader method and is better and faster then the "pre-shader support" keying method.

- **Dynamic Image** - Enables drawing a scene or part of a scene into an image. This is used mainly for scene transitions but it can be use for many other things as well. There are 2 ways of implementing it:
 - **Back buffer**: using the regular renderer but the back buffer.
 - **pBuffer**: opening another hidden "renderer"
 - The **None** option is for setup with Viz|Trio NV.

Scene default values

In this section you can set default values for several scene elements:

- **Alpha** - Sets the default alpha/keying mode.
- **Camera** - Sets the default camera range.
- **Animation Motion Path** - Sets if default animation motion path should be smooth or linear
- **Animation Motion Speed** - Sets if default animation motion speed should be manual or linear.
- **Text object Orientation** - Sets the default text orientation
- **Text object Direction** - Sets the default text direction
- **Text Object Bounding Box** - sets the default bounding box type. The new type is uses a different method for calculating the bounding box which is more effective. The old style is kept for backwards compatibly considerations.
 - **Old Style** - The old style will be used as default
 - **New Style** - The new style will be used as default.
 - **Force New in Load** - The new style will be used as default and in scenes with old style bounding boxes, new style boxes will replace the old ones.
- **Render half height** - Viz|Artist will render fields instead of full frames. Normally it renders 50 or 60 full height frames and produce twice the amount of pixels needed. This is done to achieve better quality. If you pick every second line of a full height frame and send it to the output it will look better. Antialiasing, texture filtering works "better" if you render full height. So when Viz renders half height, you will lose a little bit of quality, especially on some textures, but it nearly renders twice as fast, because pixel fill is much lower, its half in fact!

Sensor

The sensor section is relevant for programmers who write control programs. It makes it possible to send real time data to Viz|Artist.

Targa3200

See: ["Configuration parameters for the targa functionality" on page 176](#)

Video Output

If a Targa video card is installed, see ["Configuration parameters for the targa functionality" on page 176](#)

Scene building objects

Viz|Artist user interface

Enter the scene building window by creating or opening an existing scene in the scene database. The Viz|Artist main window consists of four areas:

- The rendering window is in the lower right part of the screen. It shows the scene as it will look on air.
- The scene tree is shown in the lower left window. It shows all the elements of the scene and their hierarchy.
- Databases are displayed in the upper left window
- Object editors are in the upper right window.

- When the animation stage is shown it covers the whole upper part of the screen.

Interaction with the scene builder tools is mostly based on drag and drop actions.

Scene Tree

The Scene Tree is the logical visualization of all the elements in your scene. The tree consists of containers that hold objects. Objects can be geometry, images, materials, texts, transparency functions, key functions and many more. You build a scene by dragging the construction elements, like images and graphical primitives, onto the Scene Tree. (It is also possible to drag objects directly into the rendering window). In the tree you are able to create hierarchy between objects and to group objects in logical divisions. This enables you to give many containers new properties in a single operation.



A black arrow indicates that the container has subcontainers. If the black arrow points to the right the subcontainers are not displayed. If it points down you can see the sub-containers. Click on it to change its state.

Below follows some examples of icons representing different elements in scene tree containers:



Eye icons indicate if the container and all its sub-containers are visible in the rendering window. Click on it to change.



An icon showing if the container is locked or unlocked. If the container is locked you cannot drag it around in the rendering area.



An axis icon is always present. Clicking on it opens the Transformation Editor.



An icon showing the geometry held by the container. This is either an icon of the imported object or the symbol for the primitive.



A material icon indicating that the container is drawn using a specific material. If there is no material icon this means that a standard white base color is used.



An image icon indicating that the container holds an image. There are two ways an image can be applied to a container. If the container holds geometry, the image is used for texturing the geometry. If there is no geometry, the image is displayed as it is.



An alpha icon, showing that the container holds an alpha function.



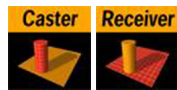
A Key icon indicates that the container also is used to generate key information.



An icon showing if a container has animated elements.



An icon showing if the container has mask operations applied to it. How to use the Scene Tree, see paragraph: ["Using Mask Function" starting on page 67](#).



These icons show if the container acts as either shadow caster or receiver. How to use the shadow function, see ["Using the Shadow function" on page 67](#)



These icons show if the container acts as either projector or projector-target. To set up and use the projector function, see ["Using the Projector function" on page 69](#)

Display Options



The 'eye icons' hide or show all containers in the tree. The "lock icons" lock or unlock all containers. Clicking the minus or plus icon will show only the uppermost containers or all containers in the tree. You can switch the tree display to low detail at the lower right of this window by toggling between the two buttons to the right. This will help you get a better overview of a complex scene tree.

Tree search

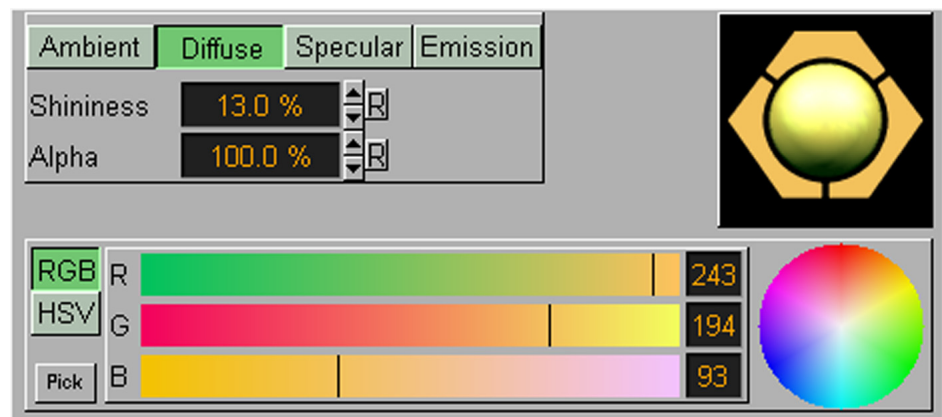
At the bottom of the scene tree window is a button labelled "search". This opens a search view which allows you to search the scene tree for folder names, objects and functions. On how to use the search view, see ["Search the scene tree" on page 51](#)

The Rendering Area

The Rendering Area shows the current state of your scene. You can to a great extent add and edit scene objects directly in the rendering area. It can be viewed through 16 different cameras with normal perspective or orthogonal views and in addition through six predefined orthogonal camera views which are top, bottom, left, right, front and back. An orthogonal camera has no perspective and therefore varying the distance from objects won't change the size of objects in the camera view. The six "side views" shows in addition the positions of cameras and light sources and give you access to alter their positions by using the mouse. These views are good for controlling the relative position and rotation of objects, lights and cameras. More details about the rendering area, see ["Working in the Rendering Area" on page 60](#)

Database editors and viewers

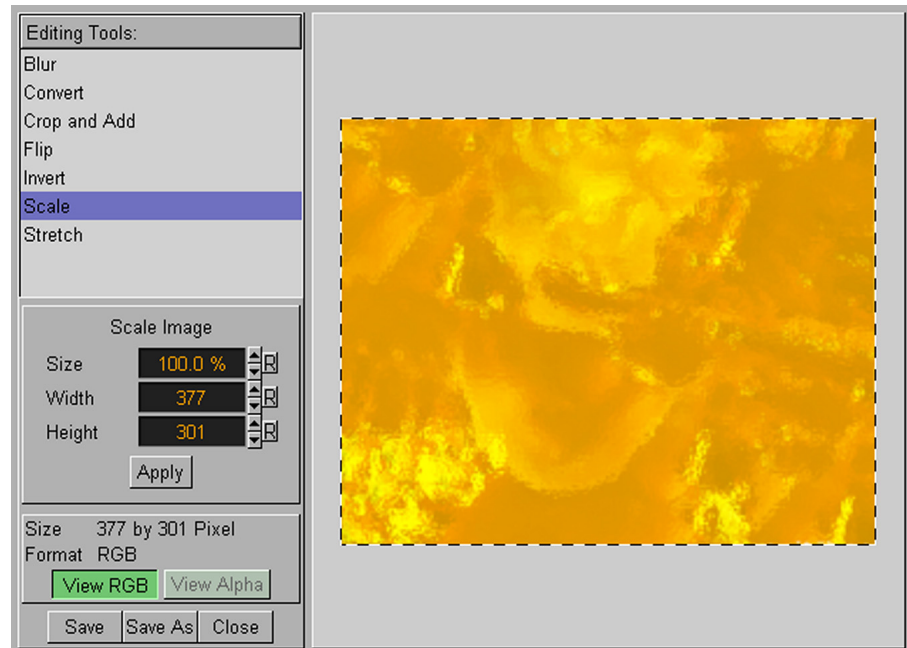
Material editor



The Material Editor opens when you double click on a material or create and open a new material in the material database. This editor allows you to edit and create materials with customized colors and light effects. Colors can be created as RGB or HSV. The functionality of the color editor you will find as a component of many other editors in Viz|Artist, the way to create a color does not differ in principle.

The editor will also open if you click on a material in a container, but it opens in the upper right window as the other container editors. The effects or types of lights to apply to a material are: Ambient, diffuse, specular and emission. In addition you can edit the containers base color. This is done by disabling the small yellow button beside the material icon in the container editor. More on use of the material functions, see ["Using the Material database" on page 45](#)

Image editor



The image editor opens when you double click on an imported image in the image database. You can view and modify images here, including:

- Scaling
- Stretch
- Crop
- Flip
- Convert format
- Invert
- Blur

All images are treated as RGB or Alpha format. If you import an image that has Alpha you can switch between viewing RGB and alpha. Modifications to an image are automatically applied to the RGB or alpha information. After changing properties you can save the image or save as a new image.

Font viewer



The font viewer opens when you click on a font in the font database. It allows you to view all elements of a font type and test some settings. In addition it allows you to define the kerning between specific character pairs. These settings can be saved. To set the kerning for a character pair, write them in the text field and drag them apart using the mouse. All the other settings like wireframe, outline and resolution is just meant as an aid for testing that you fonts has been properly imported.

Scene editors

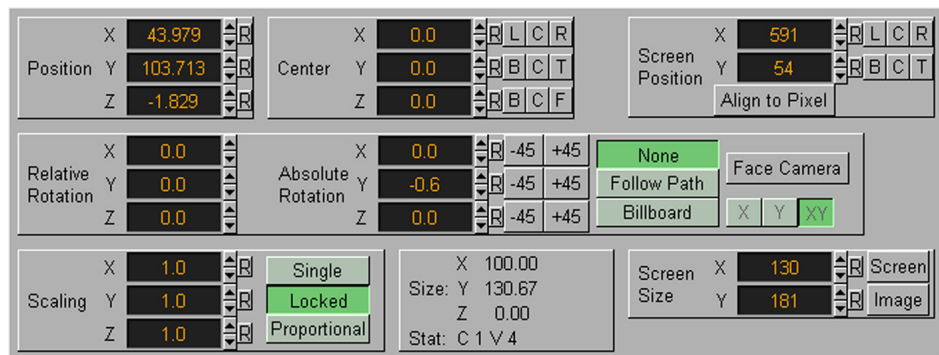
Container editor

The container editor opens when you click on the axis-icon of a container in the scene tree. It shows in the top right part of the screen. The Container editor gives you access to all the editors needed to edit all elements within a container. The "visible" and "locked" buttons do the same as the corresponding eye icons in the tree view. They hide or show an object. By clicking on an icon you get the appropriate editor. The yellow bars to the right of an icon indicate if this property is active in this container, you can activate or deactivate it in this view. You can also merge a container and its subcontainers to a compound object and you can split compound objects. More on compound and imported objects, see ["Using the Geometry editor" on page 54](#)

Picture showing two samples of the container editor



Transformation editor

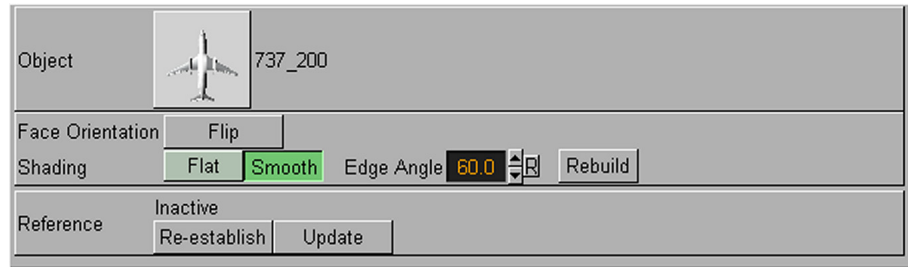


The transformation editor opens when you click in the axis icon within a container. In this editor you can change:

- Position
- Centre position
- Rotation
- Scaling

On how to use the Transformation Editor, see [“Using the Transformation editor” on page 52](#)

Geometry editor

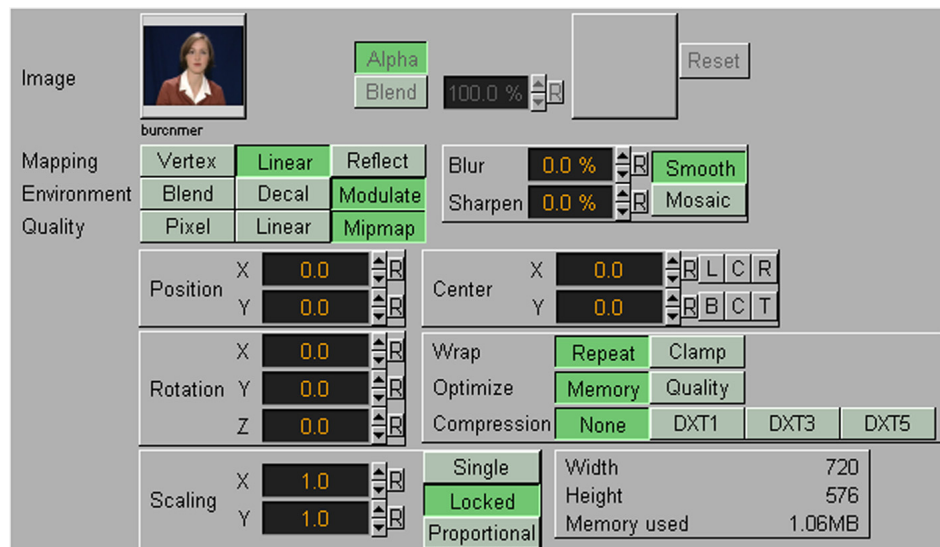


The geometry editor opens in the upper right section when you click on a icon in a container which represents a compound graphical object. This editor gives you information about some vital properties of an imported/compound object as well as its references to the object pool. When you drag a compound object out of the object pool and into the scene, a reference between the object in the database and the object in the scene is build. This means that if you change the object in the database the object in your scene is also changed, including any scene where this object is used with an established reference. If you want to have your own copy of the object in your scene you can deactivate the reference link. If you split a scene using the split-button in the container editor, the reference is automatically disabled. You can reactivate a reference that has been disabled. If the object has been split, you first have to merge it again. To reactivate a reference between a object and its source in the object pool, you either:

- **Re-establish**, which updates the container in the scene from the object in the object pool, or conversely
- **Update**, which replaces the object in the object pool with the object from the scene container.

You can in addition change the objects face orientation and the shading angle. To do this the object reference must be deactivated. More on the geometry editor, see ["Using the Geometry editor" on page 54](#)

Texture editor



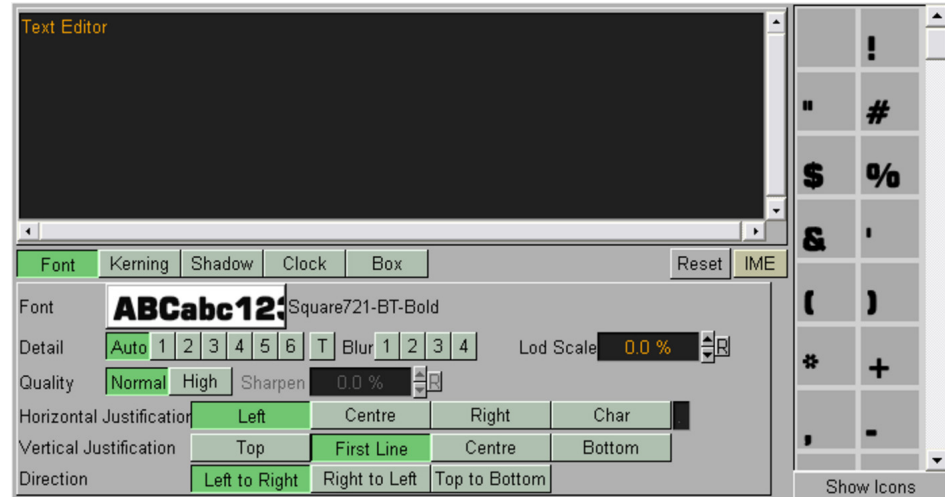
The texture editor opens when you click on an image in a container. The editor is used for editing the image texture parameters. You can alter the following properties for an image:

- Mapping
- Environment
- Quality

- Blur/Sharpen
- Position/Rotation/Center/Scaling

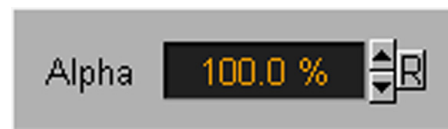
More about the Texture Editor, see [“Using the Texture Editor”](#) on page 55

Text editor



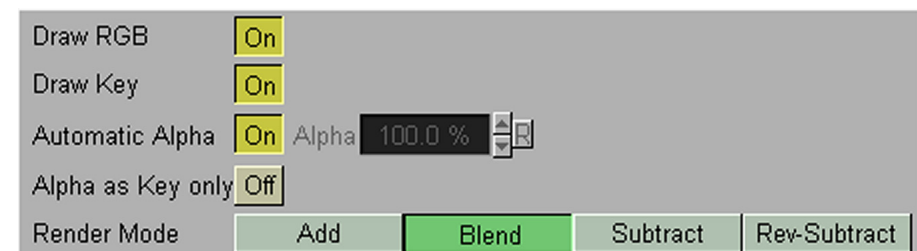
The text editor opens when you click on a text element within a container. In the text editor you can in addition to entering the text you want for the scene object, change how the text is being visualized by altering properties and adding special effects. Among the effects is a digital clock function. See more in [“Using the Text Editor”](#) on page 57

Alpha editor



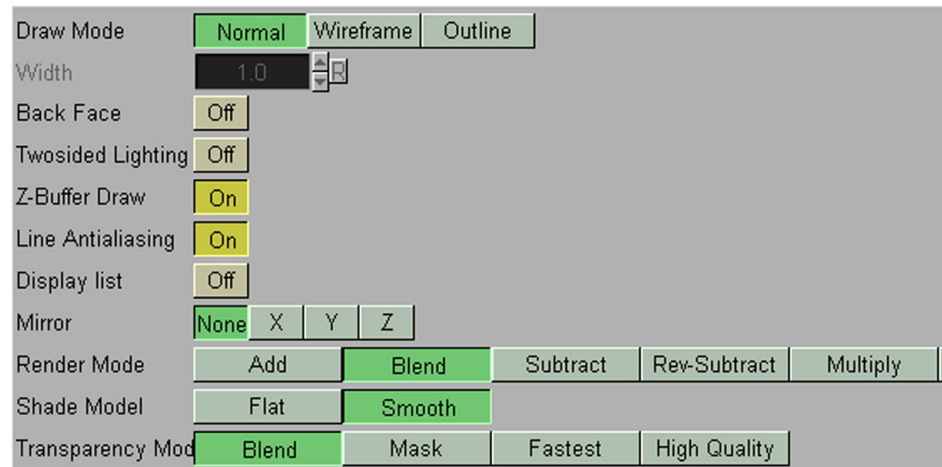
The alpha editor opens when you click on an alpha icon in a container. You use the editor to change the transparency of the container objects. If a group container has Alpha, all sub-containers will inherit it. This feature can be used to easily fade in or out a complete part of the scene-tree.

Key editor



The key editor opens in the upper right section when you click on a key-icon in a container. You can set some parameters to how the key element acts. To see explanations on use of the parameters, see [“Using the Key function”](#) on page 65.

Expert editor



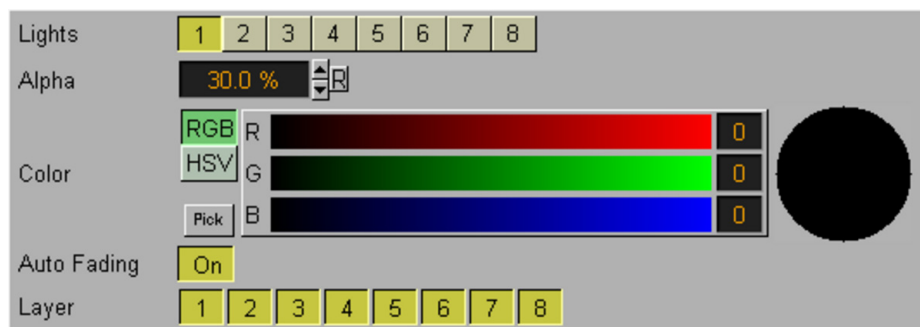
The expert editor opens when you click on an Expert icon in a container. By dragging an expert function onto a container you can set some special parameters relevant to many objects and situations. Details on the Expert editor, see ["Using the Expert Function" on page 65](#)

Mask editor



The mask editor opens when you click on a mask source- or mask target icon in a container. There are separate editors for the source mask and for the target mask, but they look the same. In these you can define which source masks who should influence different target object and conversely, which target objects who should be influenced by different source masks. How to use the mask editors, see ["Using Mask Function" on page 67](#)

Shadow Caster editor



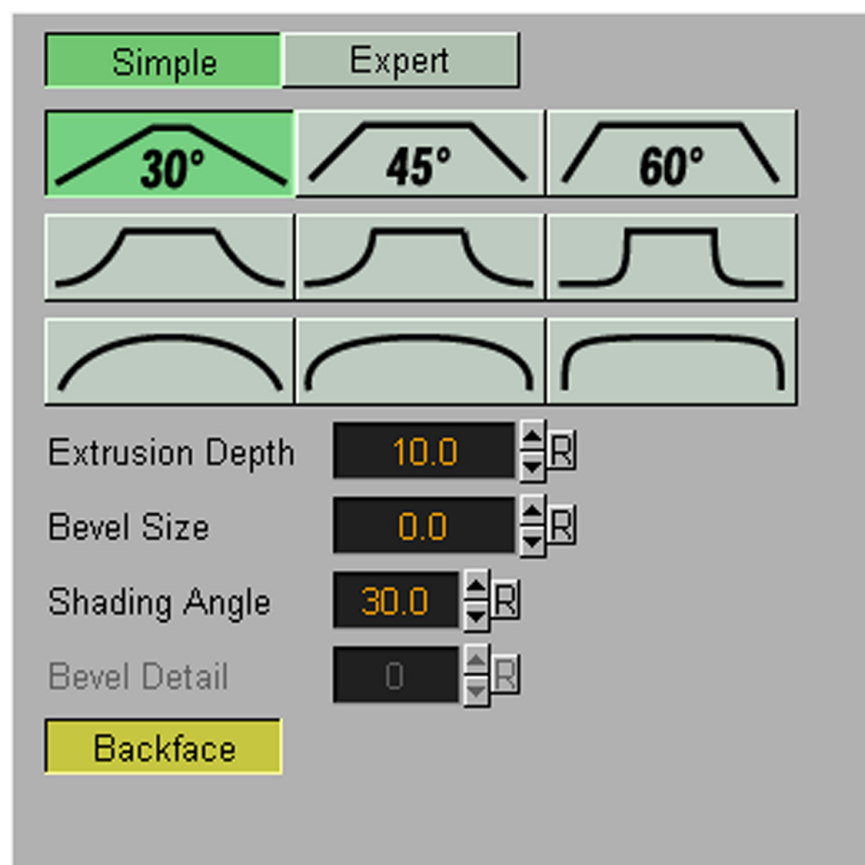
This editor opens when you click on a shadow caster function in a container. It allows you to set some parameters to how the caster object casts shadow on receiver objects. You can decide by which layers the caster will cast shadow, you can define light sources, alpha, and color. More on use and parameters, see ["Using the Shadow Caster Function" on page 68](#)

Shadow Receiver editor



The shadow receiver editor opens when you click on the icon for a Shadow Receiver function in a container. It allows you to edit some parameters to how shadow receiver objects displays shadows that have been cast from corresponding caster objects. You can decide by which layers the receiver will receive shadows, you can define alpha and enable/disable clipping and culling. More on use and parameters, see ["Using the Shadow receiver function"](#) on page 68

Extrusion editor



If you drag the extrusion function onto a 2-D primitive or a font, the Extrusion Editor opens. The editor allows you to define a bevel or extrusion type. You can change the extrusion depth, the bevel size, the shading angle and the bevel detail. More about the use and the parameters, see ["Using Extrusion function"](#) on page 67

Projector editor



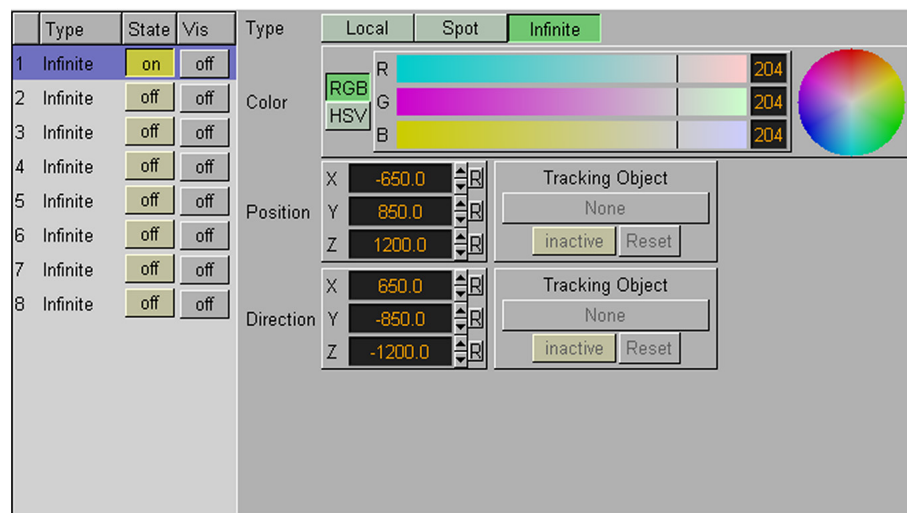
The projector editor opens when you click on a projector icon in a container. It allows you to set some properties on how it should influence containers that hold the Projector target function. On how to set up an use the projector function see, ["Using the Projector function" on page 69.](#)

Projector Receiver editor

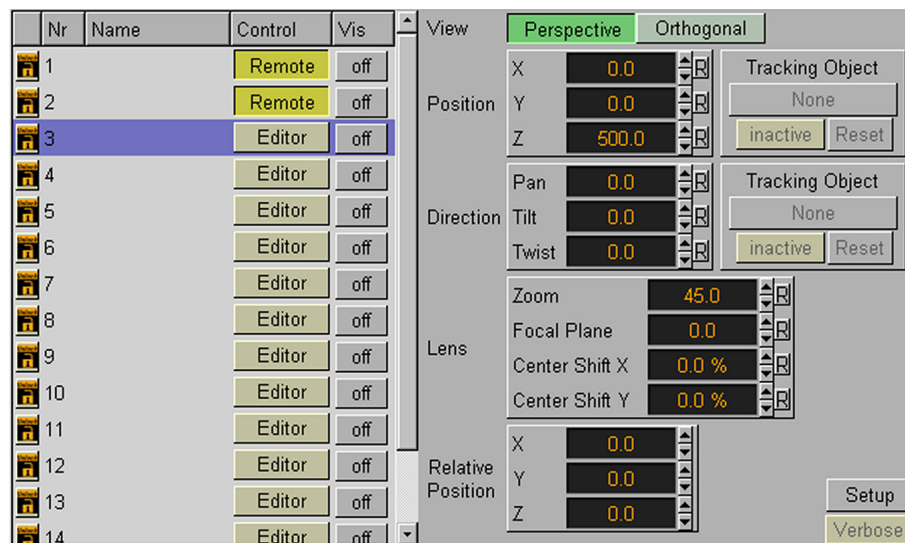


The projector receiver editor allows you to set by which layers it should be influenced by corresponding projector objects. More on the projector function, see ["Using the Projector function" on page 69](#)

Light editor



The light editor opens when you click the Light button in the main menu bar. The light editor allows you to change the light settings in the scene. You can choose between different light types, adjust the distance and angle for the light source, activate and deactivate lights and more. Light position and direction can also be altered in the side views. More on the light editor, see ["Using Light Editor" on page 70](#)

Camera editor

The camera editor opens when you click the Camera button in the main menu bar. It allows you to set the properties for the 16 virtual cameras that can be used in Viz|Artist. You can change between perspective and parallel projection. You can change position, pan, tilt, zoom and focus. You also can alter camera position and direction in the parallel views in the same manner as with lights, see ["Using the side-views" on page 64](#). If you have purchased the virtual set option, the left part of the camera editor allows you to connect to a real camera by clicking on the "Control" button and thereby changing its stage to "Remote". Tracking of cameras by objects can also be performed. You can also animate the camera in the same way as objects and lights by creating keyframes. More on use and parameters for the Cameras, see ["Using the Camera Editor" on page 71](#)

Scene views Editing in the three extra scene views

By clicking the button labelled "Views" in the main menu, three additional views of the scene opens on the left side of the screen. The views can display any camera view and is a good aid to position objects in the scene and determine the different object's positions relative to the camera.

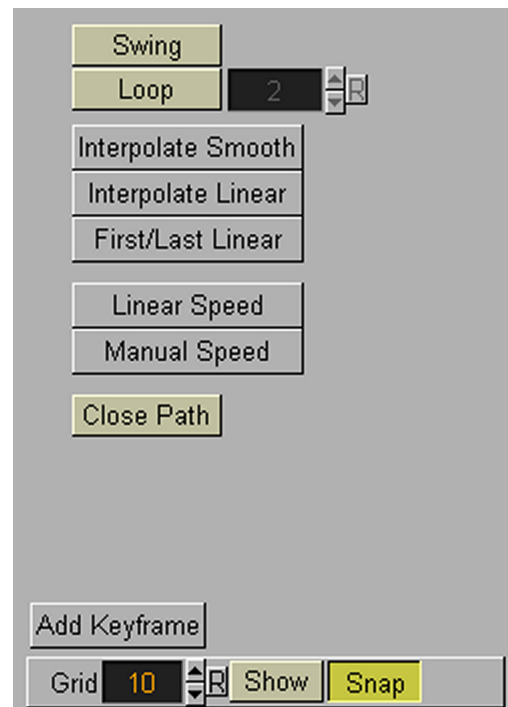
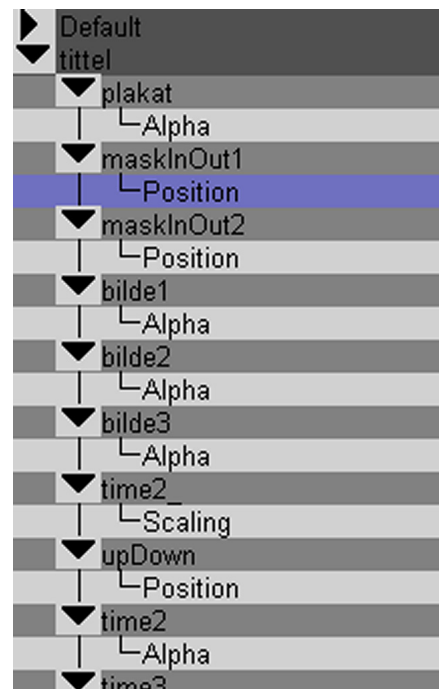
Control Channels

The control channels allows you to create named links to objects in the scene tree which makes it easier and faster for an external program to control that object. Drag a container or container property onto the control channel window. You can access this property within this container from an external source by specifying the name of the channel with a '@' in front. Be aware! If control channels are made to objects in a container tree and the tree is merged to an object, the control channels will be lost. The advantages of using control channels are:

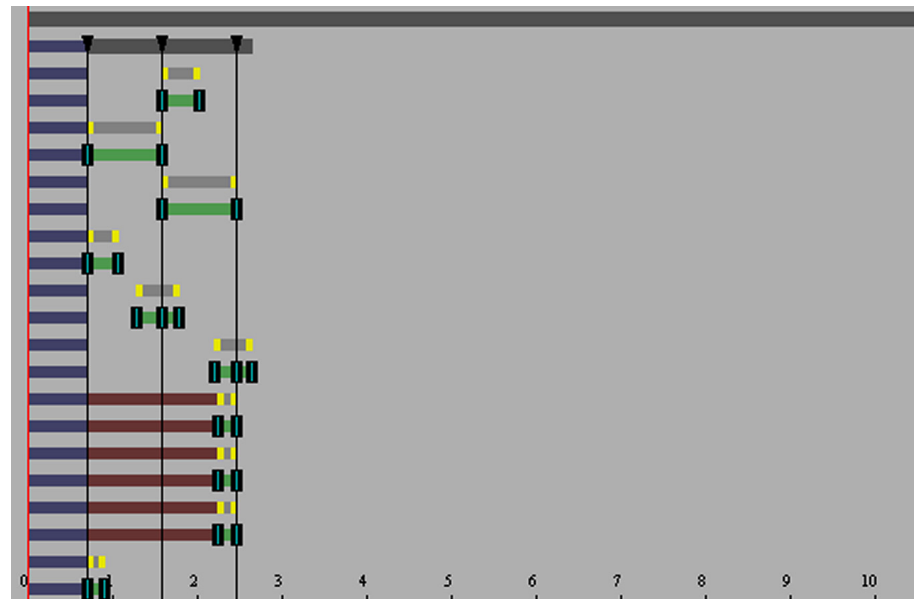
- It gives the external program quicker access to the object. Without control channels you must specify the full path to the object or the object must have an unique name.
- When using control channels, links to objects will be maintained even if objects are moved to another place in the scene tree or they are renamed.

Stage editor

The stage editor opens when you click on the Stage button in the main menu bar. The Stage Editor controls the different animation sequences in the scene. The editor is divided in three windows covering the upper third of the screen. From the left to the right are:

The channel editor view:**The director view:**

The timeline view:



- The channel editor view shows the parameter editors for the currently selected element on the stage.
- The director view shows the directors and their sub components in a tree structure. When you select an element in the tree, the relevant editor opens in the channel editor view.
- The timeline view shows the timelines for each the director, together with timelines for the objects and their corresponding animation channels. You will also find stops, keyframes and actions. A thin red vertical time slider line is always visible. You can drag this along the timelines to manually go through the animation sequence.

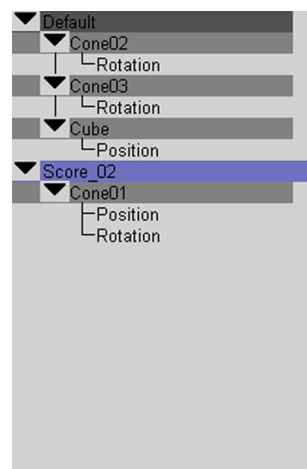
Customizing the Stage Editor

The Stage Editor have a few "guide functions", you can enable or disable:

- "Grid" displays grid lines in the Director Section. You can set the grid value which is the time between the grid lines.
- "Snap" enables the snap to grid function. If you move a keyframe or a stop tag along the timeline when you drop it, it will snap to the nearest whole grid value.

Stage directors

Picture showing the director tree with two stage directors named default and Score_02



The directors are the top level logical object on the animation stage. It can be described as a timeline manager. A director contains objects and their belonging animation channels. An object can have many different channels at the same time; a cube could for instance change position, rotate, scale, increase or decrease alpha value. Each director can hold the animation channels for as many objects as you like, but at some point adding new directors is always the better solution to maintain a well arranged structure for the animations in your scene. There is always one director present named "Default". Any object you animate is put into this director and you must move it to another director afterwards if you want another location for the object.

Stage director timeline

Each director at the stage has its own timeline drawn in a dark grey color. It shows the time scope of the director. You can add some animation elements to the director's timeline. By clicking at the top level of the director in the stage tree structure which has the same grey color as the timeline, the parameter editor belonging to the director opens in the left window. There you can add the following:

- Stops, see ["Adding stops" on page 163](#).
- Tags, see ["Adding Tags" on page 163](#)
- Actions, see ["Adding actions" on page 163](#)

In addition the editor for each director contains buttons to, start, stop, continue and reverse the animation channels that are located under that director. You can also set the director's offset which is the time it will wait before it starts any of its animations. By setting a director offset you can also delay a whole group of object animations.

Object animation timeline

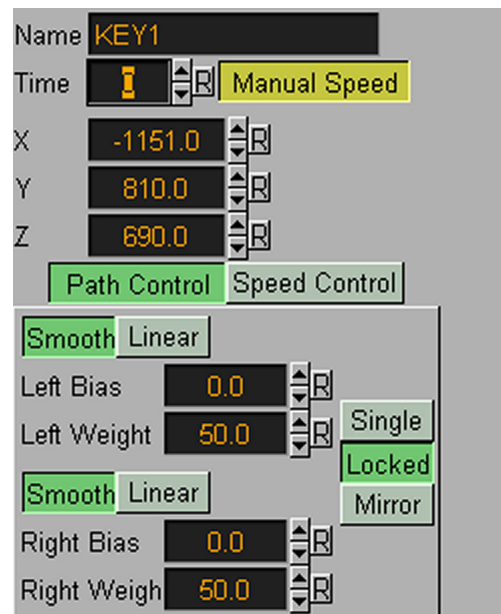
Within each stage director you will find a timeline for each animated object. The timeline is drawn in a lighter grey color than the stage director timeline and in addition it has yellow handles at the ends. Clicking by an object timeline in the director tree, opens an editor where you can set the object's time offset. The parameter is visible in the left window. This is the only parameter at the object timeline level. The offset is related to the start point of the director, and is the time the object will wait after the director have started, before it runs its own animation channels.

Animation Channel timeline

Each object can have many animation channels and at the stage there is a green timeline for each of them. By clicking on an animation channel in the

director tree, the editor, containing relevant parameters, opens in the left window. There is many possible animation channels, and the number of parameters attached, vary between different types of objects. Some animation channels get arranged in a sub group, like those belonging to a material, "ambient", "diffuse" etc. The group has a timeline attached. It looks just like the object animation timeline but is red instead. This line is meant as an extra aid to visualize the timescope of the sub group elements.

Keyframe Editor



When you animate an object by clicking the update button in the render area, changing a property and clicking update again, a set of keyframes is created. They are displayed on the animation channel timeline as a set of black squares at each side of the line, or along the line if the channel contains more than two keyframe updates. The keyframes represents the state of the object at the time update is pressed. For each update a keyframe is created. The keyframe contains the parameters necessary to describe the state of the object at the time the update is made. It also contains parameters for altering the object's movement or transformation from the last keyframe to the current and further to the next keyframe. What the parameters are, depends on the type of animation channel. For instance if it is a position channel, the parameters in each of the keyframes would be the X, Y, Z values and parameters to speed- and path interpolation. If it is a diffuse color, the values listed at each keyframe would be the R-, G- and B channel. Details on keyframe parameters, see the paragraph ["Explanation of Keyframe parameters" on page 166](#)

The Animation Channels



Almost anything can be animated in Viz|Artist. As a consequence, the number of possible animation channels is large. Some of the channel types have their unique parameters, but most of them are shared among many channels, even

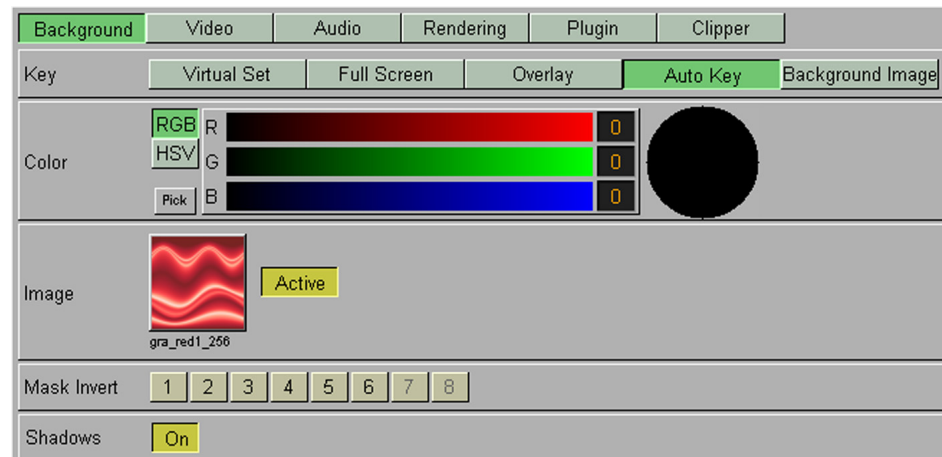
though the actual settings on each parameter are unique for each channel type. A few parameters recur on all animation channels.

In Viz|Artist you can animate:

- All numeric values relating to container elements.
- All radio buttons and check buttons, where you normally enable or disable an option.
- Colors
- Light & Camera settings

The Viz|Artist primitives can be animated on almost every value in their corresponding editor. For an overview of the possible animation channels for the primitives and objects see, ["Built in objects" on page 74](#). The explanation of the different animation channel parameters, see ["Explanation of animation channel parameters" on page 165](#)

Scene setup menu



Background menu The background menu allows you to set some basic parameters for the scene background and some global settings for the scene. You can change key settings and background color, set and define a background image, invert masks and enable/disable shadows. To view each of the parameters with their explanation, see ["Using the scene setup menu" starting on page 172](#).

Video menu The video menu contains some fundamental setup parameters for the four video input sources and for Targa video. Details see, ["Scene setup Video page" on page 173](#)

Audio menu This section lets you configure and setup the 16 available audio tracks. Details, see ["Audio" on page 179](#).

Rendering menu The rendering menu contains some basic rendering parameters. You can set Culling and depth buffer, define the Safe Area and the Title Area, enable Auto Key, set rendering to half height, set Flicker Filter, define the range for the cameras and set the aspect. To view each of the parameters with explanation, see ["Using the scene setup menu" starting on page 172](#).

plug-in menu This page is the dropzone for the scene plug-ins. You find the scene plug-ins in the function database by clicking on the button labelled "scene". Scene plug-

ins are “global” functions that have influence on the whole scene. They are mainly used when external programs are interfaced with Viz|Artist.

Clipper menu The clipper function is an alternative to the mask function. Details, see [“Using the Clipper function” on page 69](#)

Virtual Set menu This page will only be visible for those who have purchased the virtual set option for Viz|Artist. For documentation on this section, refer the manual for the virtual set option.

Device plug-ins

Viz|Artist can through its device plug-ins interface with different external devices. Some of the plug-ins allow you to receive input from the external devices, some let you send output to the devices and some let you do both. The input can be received in real time or as a part of a post rendering process. The data output to a device is post rendered except control commands, which of course are rendered in real time. To set up a device plug-in, you must enable it and set its parameters in the “Device control” section located in the config section, see [“Device Manager” on page 40](#)

Playback devices real-time and post

There are three main types of input devices that can be set up to work with Viz|Artist:

1. **Data goes through Viz|Artist core:** The data stream get processed directly by Viz|Artist. This means that the program receives the data stream from the device and renders it as an element of its own frames.
2. **Data on video channel:** Viz|Artist controls the input by giving commands to the device, e.g to start playing video, to play a certain frame or to stop playing. The data are received on a video channel, converted to texture images and then texture are rendered as elements of the scene.
3. **Control of external non-video devices:** Viz|Artist have some plug-ins that controls none video devices as a MP3-server or a video router. With this kind of devices, Viz|Artist don't receive any input directly it just triggers the payout.

In more practical terms devices that deliver input can be described as “playback-devices”. Playback can be done both in real-time and in a post rendering process. Not all devices have both real-time and post capabilities.

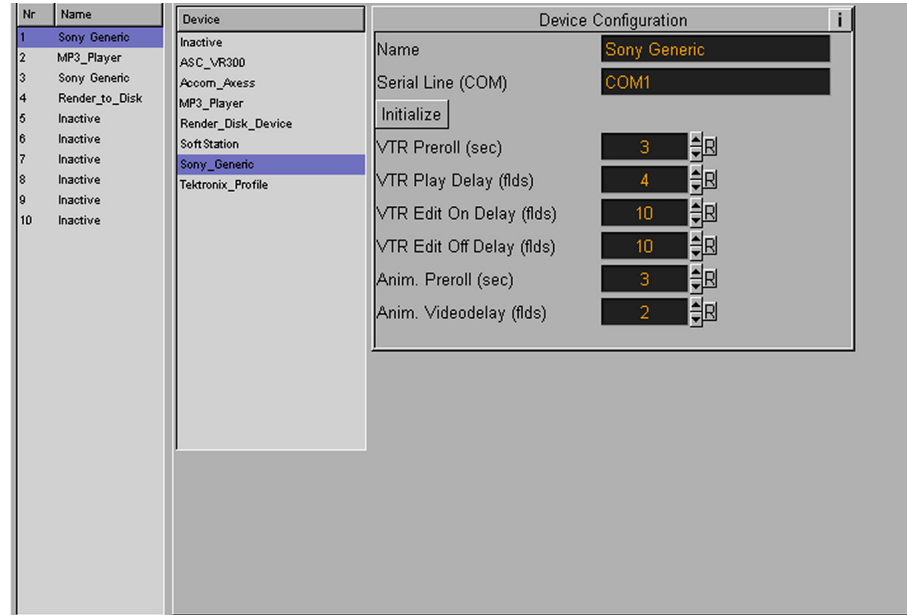
Record devices

Output to devices is normally created in a post rendering process. Post rendering gives you the possibility to render a complex scene or scene element, which is too heavy to run in real time, to a clip-station or to a hard disk in the format needed, and then replay it later with additional layers of real time graphics. Some devices have a limited range of possible function commands, and it might be that Viz|Artist through its control commands, can start, stop and cue the data stream, but nothing more. If the input from a playback device is supposed to take part in a post rendering output to another record device, a frame to frame synchronization between the video source frames and the frames that Viz|Artist renders is necessary. As a consequence, not all devices can take part as input in a post rendering process. A device with post rendering capability must have the ability to control a clip segment frame by frame available, so the Viz|Artist post rendering process can synchronize its own frames with the frames of the device.

The post rendering devices are set up in a submenu that opens when you click on the “Post” button in the main menu. This button becomes visible as soon as

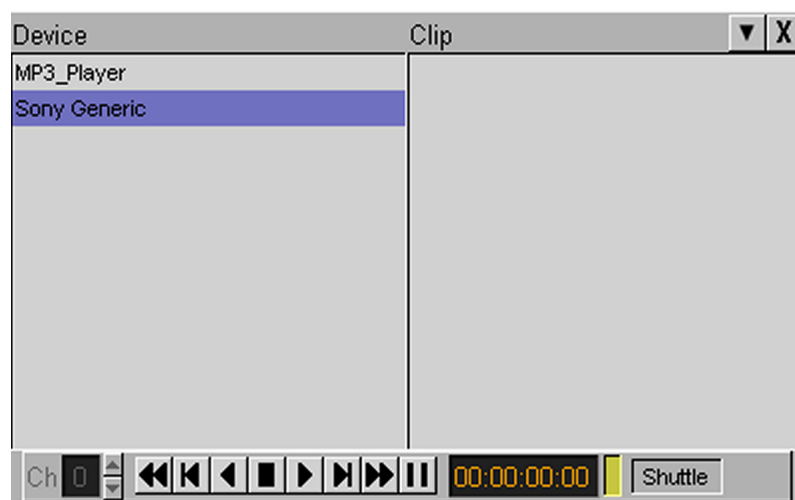
a post rendering device plug-in have been enabled. How to use the "Post rendering screen", see ["The Post Rendering Screen" on page 169](#)

Device Manager



The device manager is located in the config section. To open it, click the Config button in the main menu, and select "Device manager" in the "Sections" list. The manager opens to the right. It contains two lists, the first which have the headings "Nr" and "Name" lists up the device plug-ins currently active. The second with the heading "Device" list up all the plug-ins you have installed. As you select a plug-in, its parameter section opens to the right of the two lists. You use the manager to enable and set parameters to the plug-ins. How to perform this configuration, see ["Using the Device Manager" on page 167](#)

Device control



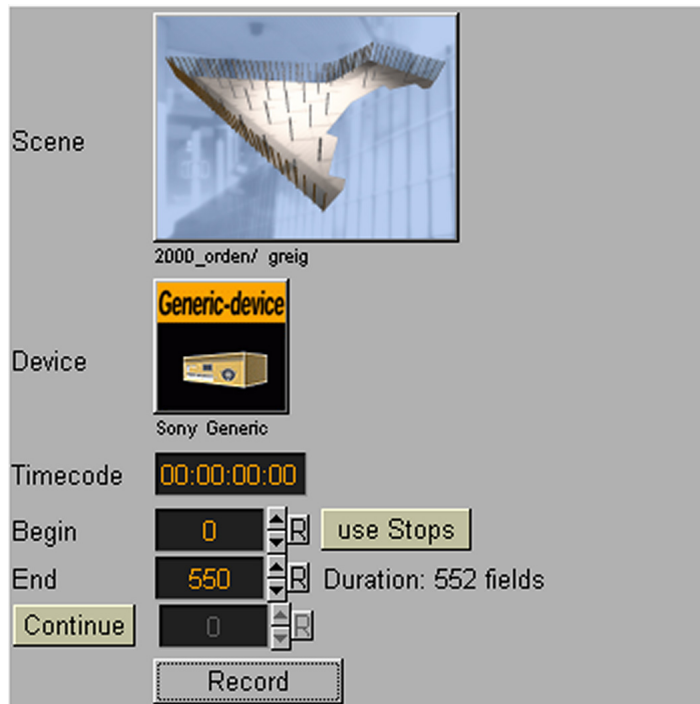
The device controller opens when you click on the button labelled "Device control" located in the lower left corner of the Viz|Artist screen. The device controller allows you to operate all the playback device plug-ins that can be added to a scene's stage. The plug-ins currently active are listed on the left side of

the device control window. On the right side you will see the clips of the plug-in currently selected on the left side, if the plug-in have clip functionality implemented. Post rendering plug-ins are controlled in the "Post" section, see "The Post Rendering Screen" on page 169. From the device controller plug-ins can be added to a stage director, a keyframe can be created and clips can be linked to the keyframe. To see more details on how to use the Device Controller, see "Using the Device Control Unit" on page 168. When in post mode, the device controller only shows device plug-ins that have post rendering capability. The plug-ins can be dragged from that list onto the post rendering window to set up a post rendering session.

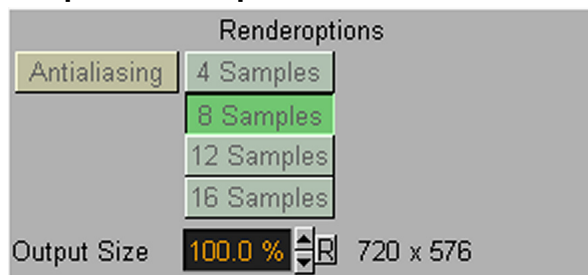
Post Rendering Screen

The post rendering screen consists of two elements:

The main post rendering setup window:



The post renderoptions:



All post rendering is controlled and set up in the post rendering screen. The editor loads the device plug-in you select and generate the post rendering output to the device. The Editor opens when you click on the Post button in the main menu. By default it renders the scene currently loaded, but you can load another scene. On how to use the Post rendering screen see, "The Post Rendering Screen" on page 169

Using Viz|Artist

Mouse clicks & shortcuts

Most value fields in Viz|Artist editors, share the same properties regarding how to enter and alter values. By limiting the number of keystroke and mouse key combinations a user must remember, quick learning is achieved. The graphical user interface in Viz|Artist is designed to provide the user with a short way to mastering. The following section applies to all editors in Viz|Artist.

Use of mouse and keyboard

Changing values

All sorts of value fields, from color to geometrical position values, they all respond the same way on mouse clicks and keystrokes:

- Clicking into a value field marks the contents to be changed by using the keyboard. You can also use the Tab key to jump to the next value field or Shift+Tab to jump to the previous.
- Dragging the mouse horizontally to the left or the right, with the left button pressed, starting inside the value field, changes the value up and down. By which increments the value changes depend on the possible range of the value in that actual field. By holding a key in addition as you drag
 - Holding the Alt-Key as you drag changes the value by a factor of 10.
 - Holding the Ctrl-Key as you drag, changes the value by a factor 0,1.
 - Holding Ctrl and Shift-Key simultaneously as you drag changes by a factor of 0,01.
 - Holding the Alt-key + Shift as you drag, changes by a factor of 100,0.
- Almost all value fields have two small arrows at the right side, click on them to increase or decrease the value. Clicking on them combined with the same key-combinations as with the horizontal dragging, the effect is the same, the increment size changes. The cursors up and down on the keyboard have the same effect.

Reset functions

Common to all editors is the reset button "R". Single clicking on, for example, "R" of a X-position resets X translation to zero. Double clicking on any of the "R" buttons of Position resets X, Y and Z position to zero, which all are parameters within the same category.

Scrolling content pages of databases

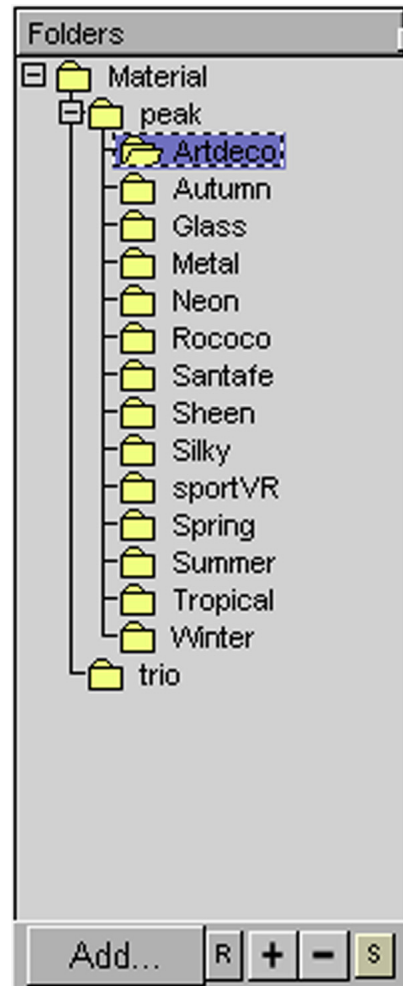
When a folder in a database contains more objects than its possible to display in the database window, you can scroll by either using the scrollbar or you can click in the window with the middle mouse button, hold it down and move the mouse up and down. Using the mouse wheel has the same effect.

Scrolling the stage timeline

To scroll along the stage timeline, hold the middle mouse button pressed and move the mouse horizontally.

Using Viz|Artist databases

The folder tree The folder tree part is common for all databases: Scene, Object, image, font, and material.



- Click the Add... button to add a new folder below the one currently selected.
- Click the R button to refresh the tree
- Click the + button to expand the tree, one node deeper per click.
- Click the - button to collapse the tree.
- The S button means "Synchronize". All pools with this option set are synchronized when a folder is selected in one pool. For instance if you click on a scene folder named "News24", all other pools with the option set will switch to the folder "News24" if it exists. Setting this option typically makes sense for combinations such as Scene and Image. Having the images for a scene in a folder with the same name as the scene folder is a common way to arrange your resources.

Using the Scene database **Editing a scene**

A double mouse click on a scene icon brings you into the scene builder modus.

Adding a Scene

Clicking "Add" creates a new empty scene. Type in a name for the scene and press enter. The new, empty scene is created in the scene database. Double click on it to load it in the scene builder modus.

Archiving a Scene

With no scene opened, by clicking the button labelled "Archive", the archive dropzone opens to the right. You can store complete scenes or parts of scenes in a file. To create an archive start with dragging the icon of one or more scenes, objects, materials, images or fonts into the archive window.

Viewing at the elements to be archived

At the top of the archive window there is a button bar which shows the number of the different element types that are loaded into the archive window. By clicking on the buttons the list below changes to show the details of the selected element. If you for instance press on the object button all objects included in the loaded scenes are displayed in the list below.

Deselect elements for archiving

By default all elements in the scenes loaded into the archive window will be saved in the archive file. In many cases you want to remove some elements from the archiving process because you already have the elements where the archive is to be imported; typically the case with fonts. Remove elements from the archiving process by dragging them onto the recycle bin below the element list. You can also remove all elements of a kind by pressing the button labelled "remove images", "remove fonts" etc. The button "remove all" clear all objects from the archive window.

Saving the archive

After having made your selection give the archive a name and save it by pressing the save button. This can be done either to a self defined location or to one of the predefined locations that have been set in the configuration section "archive paths". Every 3D object, font and image used by the scenes is copied to the file, except those you have deselected. More on archive paths, see ["Archive Paths" on page 16](#).

Deleting a Scene

A scene is deleted by dragging it to the trash can. The scene is then removed but the objects it consisted of like fonts and images, remains. They must be deleted individually if you want to discard them as well.

Using the Object database

Navigate the object database

The object database opens when you click on the object button in the main menu. Some elements of it are different when a scene is loaded as opposed to when no scene is loaded. With no scene loaded, the object database gives you access to imported and compound graphical objects. The folder structure at the left side allows you to browse through the subfolders of the database. When a scene is loaded, a button labelled "Built in" gives you access to the program's own built in primitives and standard objects in addition. For a detailed description of each of the standard objects, see ["Built in objects" on page 74](#).

Special on imported objects

If you import a compound object made with an external modelling program, and it appears with a different color on all parts, that means that the object contains different groups of sub-objects and they have no original colors.

Viz|Artist assigns random colors to all parts to make it more easy to view the different parts of the object. These you can edit separately by splitting the object with the split function in the container menu.

Use an object in a scene container

To use an object from the object database in a scene: With the left mouse button pressed, drag the object onto the container you want to use it in or directly onto the rendering window. The editor for the object automatically opens in the upper right window.

Save an object in the database

If you have made a compound object in a scene by merging some containers, you can save it in the object database by dragging the icon of the compound object from the Container Editor and onto the object database window. You will be prompted to enter a new name for the object. Enter a name and press enter to save the object.

Using the Material database

Use a material in a scene container

To use a material in a scene: With the left mouse button pressed down, drag the material onto the container you want to use it in.

Add a new material

You can add a new material by clicking "Add", type in a name and press enter. The new material is then visible in the database as a new icon. Double click on it to open the material editor so you can set the properties of the newly created material.

Edit a material

Double clicking a material or creating a new one, invokes the Material Editor. Change the color properties for a lit object by clicking "Ambient", "Diffuse", "Specular" and "Emission" and set the color values. You can choose between editing in RGB mode or HSV mode. The icon to the right shows the current state of the material. The color for a lit object is shown on the sphere within the icon, the three parts around the sphere show the plain color for an object that is not lit.

Ambient color

The ambient color scheme is used to create the color of an object when in a state where it's impossible to detect any specific light source. It is the kind of color you get when all light sources that keeps an object visible, first have bounced of many surfaces and thereby lost all directional qualities. The light surrounds the object and the color looks the same at every side of the object. Since the light does not come from any direction but instead "from all" directions, the color/ light spreads evenly at the object.

Diffuse color

The diffuse color scheme shows the objects color when illuminated by a directional light source. It spreads evenly on the object. Diffuse is the objects "core color" observed under "normal" lightening conditions. Any light normally has some level of the diffuse element.

Specular color

The specular color scheme shows the color we get when light comes from a particular direction and bounces back from the surface in a mirrored direction. The specular color is in fact the direct reflection of a directional light source.

This is typical effects of shiny metal and plastic. Specular color can be seen on as a kind of shininess color.

Emissive color

Emissive color is produced by an object's "self glow". The object itself is the light source illuminating the color. The emissive color is unaffected by any other light sources and does not produce any light for other objects in the scene.

Base color

When you click on a material in a container and then disables the material icon in the container editor by pressing the small yellow button, it shows the object's base color. This is a color without any shading and it is not influenced by light, which means that it is visible even if there are no active light sources. Its initial color is made through a conversion of the material RGB components, but you can edit the values afterwards.

Shininess

If you have set some level of specularity, the shininess defines how spread out the highlighted area is.

Alpha

The alpha value can be set directly on a material to adjust the level of transparency.

Color picker



Clicking the Pick button starts a color picker functionality. You can then click any place on the screen to pick a color, also outside the Viz|Artist GUI.

Using the Font database

Use a font in a scene container

To use a font in a scene: With the left mouse button pressed down, drag the font onto the container you want to use it in.

Editing a font in the font editor

By double clicking on a font in font database the font editor opens. This editor allows you to view a font and change some properties. The upper third of the screen shows the whole character set of the font. Clicking on a character displays it in the lower part of the window. You can also enter characters by keyboard. To enter characters by the keyboard, left click into the text field to the right of "Please Enter String". Clicking reset will blank the screen. The display will scale as you enter more characters. You can view the different resolutions that were generated at import time. These will be used to render text with automatic level of detail.

Changing font style

In the font editor, the 'Style' option allows you to view the characters either filled, outlined or in wireframe mode. Viewing in wireframe you will see how the program meshed the characters at import time.

Changing font kerning

With a character pair displayed, with the left mouse button pressed, you can click on one of the characters in the display window and drag it to the left or

right. This will change the kerning between this specific character pair for this font. When you save the changes the specific kerning will then be used when rendering text with this font. This enables on-time modification to the kerning of character pairs, saving time and ensuring consistency in use.

Using the image database

Using an imported image in a scene container

To use an image in a scene: With the left mouse button pressed, drag the image onto the container you want to use it in.

Using a non imported image

With a scene opened, click on the "File" button below the folder three. A file browser window appears and image files are listed. This view generates no thumbnails, it just lists the filenames. Select a folder containing images and drag the desired filename onto a container to use it in a scene. This is not an ideal method! The best way is to import the image first and thereafter use it in a container. This function exists mainly for testing images from Viz|Content Pilot's image pool.

Exporting an image

You can export an image by dragging it to the export drop zone on the bottom bar of the image database view. You can export in SGI .rgb, jpeg (jfif) or alias format.

- Drag the image to the export drop zone
- Select the folder where you want the image to be exported to.
- Select the format you want for the exported image.
- Click OK to export

Grabbing a video still

To grab a still from a video source, click on the "built in" button and the video sources appear. Double click a video icon and grab a still from the live video input. There are 4 live Video Inputs available depending on the Hardware you use. To configure the input video formats change to the config menu - video input settings. See ["Video Input" starting on page 19](#).

Combine a RGB and an Alpha image

This function is only available when no scene is loaded. To create a mixed image of a RGB and an alpha image, drag a RGB image and an alpha image to the corresponding RGB- and Alpha icon at the bottom bar. Thereafter press the "Combine" key and give the new image a name.

Using the Function Database

Built in standard functions

To add alpha, key or the mask property to a container you just click the icon and drag it onto the container in the tree or directly onto the object in the rendering area.

Container plug-ins

The container plug-ins are located below the standard functions in the Function Database, and are visible when the container button is pressed. To use a container plug-in, drag it onto a container. The editor opens when you thereafter click on the plug-in symbol in the container. To see detailed description of the container plug-ins, see ["Container plug-in functions" on page 128](#)

Scene plug-ins

Scene plug-ins are located below the standard functions in the Function Database and are visible when the Scene button in the database is pressed. To use

a Scene plug-in, open the setup menu and select the page named "plug-in". This page contains a dropzone for the scene plug-ins. Drag the selected plug-in from the database and onto the dropzone. The editor opens to the right of the dropzone.

Shader plug-ins

The Shader category contains folders with different effect plugins that use the pixel and texture shader technology that is found on newer graphics boards. To see a detailed description of the shader plug-ins, see ["Shader Function plug-ins" on page 155](#)

Viz|Artist Export and Import

Exporting Exporting in Viz|Artist is done from the "Scene archive" function, see ["Archiving a Scene" starting on page 44](#).

Importing **Open the import view**

You open the import view by clicking on the import button in the main menu bar. Import can be done from disk/network or from a FTP-server.

Browsing the network or disk for files to import

Click on the "Disk" button to set the import view in disk/network mode. A double click on a directory opens this directory. A single click on a directory displays the contents in the right window. Select one or more objects in the right window in a normal manner by clicking on them with the mouse cursor. Select a "block" with the SHIFT-key pressed and select multiple items with the CTRL-key pressed. Press the Expand archives button to expand or collapse the archives.

Connecting and browsing a FTP-server for files to import

By clicking on the "FTP" button the import view switches to FTP-mode. Connect to the wanted FTP-server by entering a connect-string in the text field on top of the window or double click one of the FTP-Aliases in the lower left window. The FTP aliases can be defined in the config section, see ["Default FTP URLs" on page 16](#). A manually entered connect string must have the following format: "uid:pwd@hostname". E.g. "Artistuser:mypassword@ftp.ourpft-server.com"

Importing the selected files

Import one object by double clicking it. Import more than one object by selecting them and clicking "Import".

Viz|Artist can import fonts, images, 3D objects, materials and texture vertices.

Import supported font types

- Adobe Type1 fonts (.pfb files)
- Adobe Type2 fonts
- True type fonts (recommended) most common format.

Import supported image formats

- SGI's .rgb
- .rgba
- .bw
- .gif'

- .tif
- .tga
- .ppm
- .tex
- .txc
- .psd (Photoshop, Layers defined in the Photoshop file will be imported as separate images into the image database).
- .ai - Adobe Illustrator files up to version 7.0 can be imported in the polygon editor. Add a polygon to your scene, open the editor and press import. Only polygon outlines are supported. Polygon object, see ["Polygon" on page 75](#)

Import supported 3D formats

- Wavefront .obj.
- 3Dstudio .3Ds.
- Autocad .dxf
- VRML 2.0
- Softimage

(All objects that are to be imported into Viz|Artist must be "polygon objects").

Import supported materials and texture vertices

- .obj
- .3Ds

Common problems with importing

- Objects have the wrong light settings. The reason for this might be that the polygons face orientation is wrong. To fix the problem flip the polygons in the geometry editor.
- Objects have the wrong shading or they look rougher than when it was made in the external program. The reason for this might be that the shading angle at import was not set right. To fix this, try changing the shading angle.
- You lose texture coordinates as you import or they are wrong. The cause for this might be that you exported the object in a texture format that did not support texture coordinates or you have forgotten to export the texture coordinates. The solution might be to change format and/or apply the correct settings as you export from the modelling program.
- An object looks exploded with the elements having moved randomly apart. The reason for this might be that some techniques are used in the external 3D studio program that are not included in the Viz import, for instance "null groups". The solution for this is to avoid using these functions when working on objects that are to be imported into Viz|Artist. Hint: To make sure that the imported objects maintain their correct relative position and that they keep the correct scaling, do the following in 3D max: Select the objects that should be exported, from the utilities menu, select resetXform\Reset selected and to finish, from the file menu, select export selected. In LightWave always remember to triple the objects before you export. That means that the objects must be built by triangles and not by polygons with more than three sides. Tripling is a function in Lightwave
- You lose special characters when importing open type fonts. The solution for this is to convert open type fonts with a font tool like "Fontographer" before importing them.

Using scene building objects

Using the scene tree

Building, navigating & editing the scene tree

Clicking on an icon within a container has the following effect:

- The container is marked in the tree.
- Within the rendering window a bounding box is shown for this container (Assuming the BB-button by the rendering window is enabled).
- In the upper right part of the screen an editor is shown.

To add a container you drag an object from the database into the tree or into the rendering window. If you drag it into the tree you will see an arrow in the already existing containers that shows you where the new container will be placed.

- If the arrow points up as you release the mouse button the container will be placed above the container.
- If it points down, it will be placed below the container with the arrow symbol.
- If it points to the right, it will be placed as a subcontainer of the container which shows the arrow symbol.

So if you for instance click on the material icon of the container the container will be marked and the material editor opens. If you then click on the name icon of another container, this container is marked and the material editor is opened if it holds material. If not, the transformation editor is shown.

Unmark a container

To unmark a container click on the black background of the rendering window or press the space bar while the cursor is placed in the rendering window. The bounding box will then disappear from the selected object. The container is still colored blue.

Edit a container name

Double click on the name icon of a container to edit its name. Press return when you are done or Escape if you don't want to change. After having changed the name and still being in editing mode (enter is not pressed yet), you can traverse through the tree with Cursor down and Cursor up if you want to edit another container name. Holding down the CTRL key as you right click on a container will also set a container in name editing mode.

Move a container

To move a container to another position in the tree just drag it with the left mouse button pressed and drop it at the desired location. Make sure that you drag the "name-section" of the container and not one of the container properties.

Copy a container

To copy a container hold down the Ctrl-key while dragging with the left mouse button pressed. Release the left mouse button when you are at the wanted location in the tree. To copy multiple containers, first select them and perform the same action as with a single container.

Create multiple copies

To create multiple copies of a container or a selection, hold down the CTRL-key while dragging with the left mouse button pressed. Keep holding it pressed and click the right mouse button when the cursor is at the drop position. Each time you click a new copy will be made.

Change an object

To change an object within a container, drag an object from the database onto the container. The old object will be replaced by the new one.

Drag the same object to multiple containers

To drag the same object into more than one container, drag it onto the first container and drop it by pressing the right mouse button once without releasing the left one. Then go on to the next container, press right mouse button to drop and so on. You can also select all the containers you want the object in and thereafter drag the object onto the selection.

Drag an object directly to the rendering window

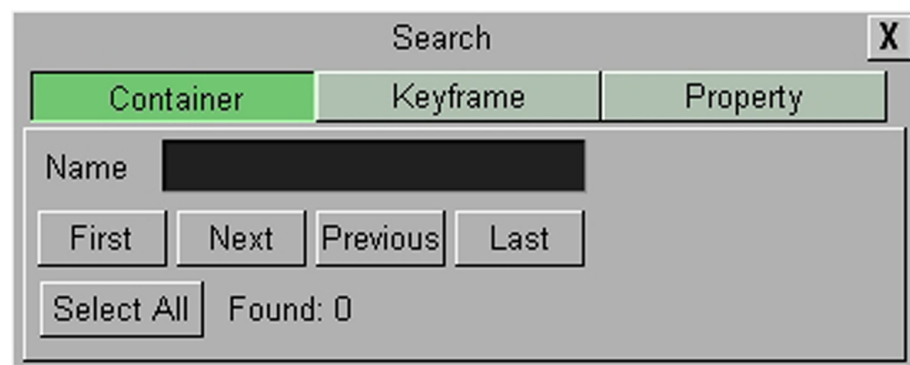
You can also drag objects out from the database and drop them directly onto the rendering window. This will add a new container if you drop it onto the background or change an existing container if you drop it onto this container. To show that your drop action will change an existing container its bounding box will change its color.

Delete a container

To delete a container click its name icon and drag it into the trashcan at the lower left of the screen.

Delete an object within a container

To delete an element loaded in a container, drag it into the trashcan. You can also drag the same object type from multiple containers. Select the containers and drag the object from one of them onto the trashcan.

Search the scene tree

The search view opens when you click on the search button which is located in the lower left corner of the tree window. The search function allows you to search for elements in the scene tree using some different criteria:

- **Container**, allows you to search for containers with specific names.
- **Keyframe**, allows you to search for keyframes that you previously have named.
- **Property**, allows you to drag a property, e.g. an alpha function or a cone object, onto the dropzone in the search box. The search then finds all containers in the tree with that property.

Every search that returns a result with more than one hit, starts showing the first one found. (The search starts from the top of the tree). You can jump between the containers that matched the search criteria with the four buttons below the name field, named "First", "Next", "Previous", "Last". The button "Select All" selects all containers that matched the search so you can edit them simultaneously

Using the Container Editor

Invoke an editor for an object in a container

The Container editor opens when you click on a container. It shows icons for all the elements in that container. To show the actual editor for an object within a container, just click on the object icon, and the editor opens to the right.

Hide/show container or objects within a container

To hide or show the whole container, click on the hide/show buttons. This have the same effect as clicking the hide/show icon on the corresponding container in the rendering tree. To hide or show an object within a container, click on the small yellow button to the right of the object icon.

Delete objects or functions from a container

To delete elements or functions from a container drag their corresponding icons onto the trashcan.

Split or merge a container

The "Split" button splits a compound container into its subcontainers. If the marked container already is split there is a "Merge" button instead. By clicking this button you can merge all sub-containers into a compound single object.

Save a compound object.

A container with subcontainers that have been merged by clicking the merge button, constitutes a compound object. To save this in the object database, drag its icon from the container editor and onto the object database. You can also drag the object icon from the container in the scene tree, but be aware, drag the object icon, not the whole container. Having dragged the icon onto the object database, you will be prompted to enter a name for the object by which it is to be stored in the object pool. A saved object can be used in other scenes and will per default have an active reference between the stored object and all scenes where it has been used. By planned use of stored objects with active references, you can make changes to many scenes at the same time, which will typically be a meaningful way to work when you design a set of scenes that share many design elements.

Using the Transformation editor

General description

The transformation editor allows you to edit the basic geometrical properties of the objects, like size, position, rotation and position of geometrical centre-axis.

Position an object

You can position an object in the transformation editor by entering values for the X-, Y-, and Z-position.

Scale an object

Scaling can be done in the following manners:

- **Single**, each axis by itself.
- **Locked**, so X, Y and Z scaling will have the same value.
- **Proportional**, adds or subtracts a proportional value on each axis.

Rotate an object

- **Relative Rotation**, rotates the object around one or more of its own axes. The degree of rotation is calculated from the objects own rotation coordinates. What you enter as relative rotation will be visible as an addition or subtraction on the values for the absolute rotation. With the

bounding box option enabled, the objects rotation axes will be visible as three lines, a red, green and blue.

- **Absolute Rotation**, rotates the object around one or more of the parent container's coordinates. With no parent container the axes will be the scene's "world coordinates". When below a parent container, the axes will be relative to the rotated position of the parent container because it inherits the rotation values from its parent or parents. This is important to remember when you have many sub-container levels.

Set the geometrical centre-axis

- **Center**, will move the axis origin of the object. You can put it to the left, centred or the right of the objects X-axis by clicking "L" or "C" or "R". Or put it to the bottom, centred or top on the Y-axis by clicking "B" or "C" or "T" or to the back, centre or front on the Z-axis by clicking "B" or "C" or "F". The rotation axes of the object will be visible in the rendering window if the BB-option in the button bar by the rendering window is enabled.

Screen position set by pixels

In the upper right corner of the editor you can set the screen position for images and some objects by specifying pixels instead of the normal way by using the position editor that does not use pixels as input values. To be able to do this, the object/image must face the camera, click the Face Camera button to achieve this. When you do changes in the Screen Position Editor, the X and Y values in the position editor will change. You can use the buttons L, C, R - which is Left, Centre, Right and B, C and T which is Bottom Centre and Top, to position the image/object. The positioning is performed based in the current Z-value for the container as well as the current camera view. If you for instance enters the pixel values X=0 and Y=0, that will place the object in the upper left corner, and thereafter changes the Z position, the object will not align the viewport corner anymore. You then have to set the pixel values to zero again to reposition the image/object.

Align to pixel

When you use the normal position editor to position an image/object that is facing the camera head on, there often appear cases where the image is off by fractions of a pixel. This can happen because of rounding errors in the "screen size" and "face camera" functions. This can produce visual artifacts in some cases on some graphics cards. When the image/object is not aligned to the screen pixels, the "align to pixel" button will become red. If you press the button the image/object will be moved to the nearest matching pixel.

Screen size

In the lower right corner is an editor that allows you to scale an image that faces the camera, based on pixels. The Screen button scales the image' to the size of the screen. The Image button scales the image to images original pixel value.

Follow functions

The buttons to the right of the Absolute rotation values allows you to define how an object relates itself to the camera.

- **Follow path**. With this option enabled, an animated object will rotate itself, as it follows the animation path, so the Z axis of the object is aligned with the path as the object moves.
- **Billboard**. With this option enabled, the container will keep the selected axis or axes in a frontal position relative to the camera as the camera moves.

Face Camera

The Face Camera button rotates an image/object to face the camera.

The object properties box

Down to the right you can see the object properties.

- **Object size:** shows the real size of the object in centimeters.
- **Stat**
 - **Containers:** shows the number of containers that are selected. If a container with subcontainers is selected, all sub-containers are counted in addition to the main container. A container with an object is counted as 2.
 - **Vertices:** shows the number of vertices that are rendered within the selected container (including all sub-containers).

Transferring properties from one container to another

Using a simple drag and drop function, it is possible to transfer properties like, position and rotation, from one container to another. If a container has a certain position and you want one or more of the other containers in the scene to have the same position you can do the following:

- Open the transformation editor of container you want to transfer the properties from. This container is then selected and marked with a blue color. With the left mouse button and the Shift- or CTRL-key pressed, select all the containers you want to give the new properties.
- Transfer thereafter the propriety you want to distribute onto the selection, by dragging the whole property object from the transformation editor and drop it onto the selected containers. If it's done correctly, you will see an icon with the property name as you drag it along.

This way of distributing transformation properties, can be done with position, rotation, scaling and center.

You can also transfer properties between containers by dragging functions and material icons and objects. If a container for instance has an alpha function with the value 70% set in the alpha editor, both the function and the value will be transferred when you drag it onto another container.

Using the Geometry editor

Object references

If you use an imported or compound object in a scene, there is a reference between the object in the scene and the object in the database. If you make changes to the object in the scene the object in the database changes to. It is in reality the same object unless you reactivate the reference and thereby create a unique object for you scene. Keeping the external reference active is a good idea if you create a set of scenes with the same theme. If for instance all the scenes should have the same background and the background element is created as an object with active reference, you can make changes that will apply to all the scenes using that object. You then save the time it takes to change the individual backgrounds in all scenes if that where the case.

Deactivating a geometry reference

To deactivate an object reference, click the deactivate button in the geometry editor. The reference is automatically deactivated if you split the object. Changes you now make to the object in the scene will after this not lead to changes in the original object in the object pool. To make this happen you must reactivate the reference

Reactivating a reference

With an inactive reference there are two different ways to make it a reference again:

- Clicking "Re-establish" will change the object in the scene to the object in the pool.
- Clicking "Update" will overwrite the object in the pool with the object in the scene.

You can click the object icon in the Geometry Editor and the object will be selected in the object database.

Shading

Imported polygon objects normally use shading for creating a smooth surface on the object which is set together by a planar polygon mesh. The shading value in the geometry editor sets a threshold between which edges those are to be smoothed by shading and which are to be kept as sharp edges. All edges with an angle smaller than the defined value are smoothed by shading. Shading in Viz|Artist are performed in the following way:

- A shade value is produced at each vertex of the polygonal mesh representing the surface of the object. This is done by creating a surface normal on each polygonal facet. The surface normal at each vertex is created by averaging the surface normals for the surrounding facets. The shade at each vertex can then be calculated.
- Once the shade at the vertices of the polygonal mesh are known the shade at points interior to the polygonal facets are interpolated from the values at the vertices.

This technique makes curved surfaces look 'smooth shaded' even though based on a representation of planar facets. The interpolation of shade values is incorporated into the polygon scan conversion routine. Hence an increase in realism is obtained at far less performance cost than carrying out a pixel-by-pixel shading calculation over the whole original surface.

By pressing the "Flat" button all shading are disabled.

Face orientation

After importing objects made by an external 3D program, it might happen that the polygons face orientation is wrong. By pressing the flip button the polygons are set correctly.

Using the Texture Editor

Editing the texture properties of an image

To add a texture to an object, just drag the image onto the container containing the object, a small icon showing the image is created in the container and the image is textured on the object. The Texture Editor opens in the upper right window by clicking on the texture icon in the container. At the top you see an icon of the image. Clicking this icon will select the image in the image database.

Mapping

The mapping section adjusts the way texture mapping is done.

- **Vertex**; means that the texture mapping is done using the objects texture- or UV-coordinates. These coordinates index a texture image, and are interpolated across the object to determine, at each of the object's pixels, a texture image value. In most cases on 3D objects, this will give the best looking result. The primitives in Viz|Artist have defined their texture coordinates. Imported objects can have texture coordinates if they have been generated by the external 3D modelling program.
- **Linear**; is a "straight forward" mapping where the image so to say is laid over the object directly on the frontal plane. The object is then moved through the image and every pixel on the object inherits the pixel it "touches" on the image. On 3D objects this will cause a repetition of the pixels on the surfaces in the Z-axis. If you lay a picture on a cube, the front side and backside will look ok, but the remaining sides will be a repetition of the pixels laying on the edge of the front side and the backside.

- **Reflect**; maps the texture as if it were a reflection from a surrounding object. The reflect setting simulates that the texture image is an all-embracing object that casts its reflection on the object. The object acts in a way like a mirror.

RGBA information

The Environment parameters specify the way the texture's RGBA information is combined with the object's RGBA information.

- **Blend**; blends the RGBA parameters of texture and object.
- **Decal**; uses only the texture's RGBA information and disregards the objects RGBA.
- **Modulate**; uses the texture's RGBA information and enables in addition use of lighting on the textured object.

Quality

The Quality parameters specify a trade between rendering speed and texture quality.

- **Pixel**; shows the texture "As it is" with no smoothing and antialiasing. It takes little resources to render but it can often create a poor visualization. This can be visible as "toothed" lines and transitions, especially when close to the texture. A motion of either object or camera, especially in the Z-axis, will create a lot "blurring motion" of the texture as the texture scales to fit the object.
- **Linear**; performs a linear interpolation to smooth the texture when being magnified and shrunk. The texture looks good but some distortions can be visible when the textured object is animated further away on the Z-axis. As the object then gets smaller and smaller, the shrinking and interpolation of the texture creates a lot of "noise" on the texture. As a consequence of this one can say that the linear quality appropriate when the objects that have the texture don't change their size to much.
- **Mipmap**; performs a linear interpolation to smooth the texture. It offers in addition a solution to the problem that appears on the two above when the object is being moved away in Z or shrunk. To avoid the "noise" that we see when a texture constantly scales to try to fit onto a shrinking/magnifying object, the mipmap quality creates a set of pre filtered images in a decreasing scale down to almost on pixel. These are used as the object scales up and down and a more static look of the texture is thereby achieved. Mipmap quality absorbs more memory performance than the two other qualities.

Blur and sharpen

When the mipmap quality is selected, you can set a sharpening- or blur filter. The blur filter can be set to create either a smoothing or a mosaic. The sharpen option is useful when the rendering window is scaled to "half height mode".

Position of texture

You can edit:

- **Position**
- **Rotation**
- **Scaling**
- **Centre**

When you position or scale the texture, you can specify if the texture is to be **repeated** or **clamped** ("Wrap"). Repeat means that the whole texture is repeated over again to fill out the object. Clamp means that the pixels on the end of the image are repeated to fill out the object.

Texture optimizing and compression

- **Optimize** - You can choose between optimizing for Memory or Quality

- **Compression** - This option is only relevant if you have one of the Vizrt approved graphics cards in the Nvidia Quadro family. DXT1, DXT3 and DXT5 are different compression algorithms and compress textures by 1:4 or 1:8. Setting one of these compressions saves you a lot of texture memory and increases rendering speed, but costs you some texture quality. The compression is set per image, so you can not have the same image in you scene both compressed and uncompressed. The info field in the texture editor will show you the size of the image and the memory used by it. The performance meter will show you the total amount of texture memory allocated on the board (TMS) and how much being allocated by all scenes in memory (TMC), see details about the performance meter in the paragraph: ["The left side button bar" on page 61.](#)

Use of two images in one container

You can use two images in one container. The left icon in the editor shows the first and default image. If you drag a second image to the drop zone near the "Reset" button you are provided with additional features.

- **Alpha**: A second alpha image can act as the alpha channel for the main image. This is in principle the same as the image combine function in the image database, see ["Combine a RGB and an Alpha image" on page 47](#)
- **Blend**; blends between the two images. This only works on SGI's Infinite Reality.
- **Reset**; clears the second image.

Using the Text Editor

Editing text properties

At the top you can type in the required text. The buttons named Font, Kerning and Shadow let you edit the text properties.

IME button

Clicking the IME button sets the editor in IME mode. See, ["IME button" on page 57](#)

The Font properties



The Font button allows you to edit the font in use. You can change it by dragging another font out of the font database and onto the font dropzone.

- **Detail**: Sets the level of detail, or tessellation of the font, to either automatic level of detail or 6 different fixed settings. The fonts are rendered as polygons so you can switch to the wireframe view to see the difference as you change the detail level. 1 is the highest level of detail, 6 is the lowest. If you set a fixed level of detail, the automatic level of detail (auto) will be overridden and a font will be rendered with the same level of detail independent of the fonts size. The button labelled "T" creates the font as a texture instead of as a polygon. This can to some degree create a poorer look, but it reduces the number of polygons in your scene. The texture resources will of course get an increased demand, so to render the fonts as texture is a "trade off" between polygon and texture performance. If you want to use "half height mode" rendering, then texture fonts are normally the better choice. When texture-fonts are selected, the sharpen parameter is enabled.

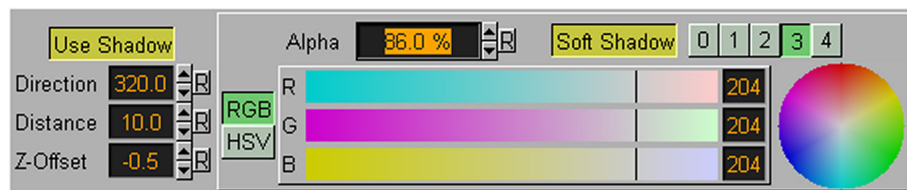
- **Blur:** Renders the font as a texture and sets a blurring filter. Select between four different degrees of blurring, degree 4 blurs the font most.
- **Quality:** This only works on machines with full scene antialiasing (Onyx, Inca). When high quality is selected it renders the geometry of the font and in addition to that an outline. The result is a very nice antialiasing, but it is a bit slower.
- **Horizontal Justification:**
 - **Left, Centre, Right,** sets the justification in these positions.
 - **Char,** allows you to define a character, typically a decimal separator, like comma, and justifies the text aligned after this character. Enter the justification character in the small text field to the right of the char-button.
- **Vertical Justification,** choose between:
 - **Top,** sets the vertical justification on the top of the whole text block.
 - **First line,** sets the vertical justification below the first line.
 - **Center,** sets the vertical justification at the centre of the middle line in the text block.
 - **Bottom,** sets the vertical justification at the bottom of the text block.
- **Direction,** choose between:
 - **Left to Right,** writes the text left to right.
 - **Right to Left,** writes the text right to left.
 - **Top to Bottom,** writes the text from top to bottom.

The Kerning properties



- **Kerning;** changes the kerning between all characters by the same amount. This is in addition to the character pair kerning selected in the Font Editor.
- **Line spacing,** modifies the distance between the text lines.
- **Word spacing,** changes the size of the blanks between the words in the text.

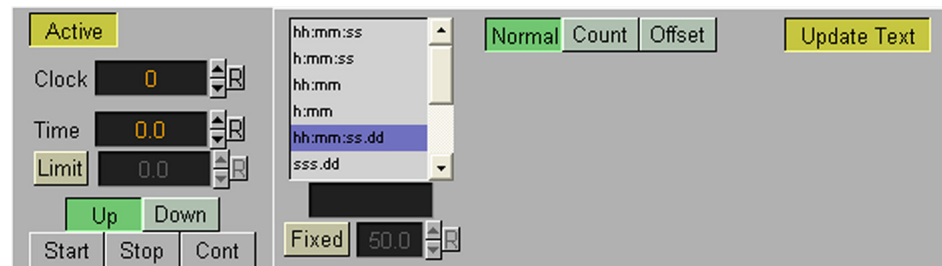
Shadow properties



Shadow; adds a drop shadow to the font. Click the "Use Shadow" button to enable shadows for the font. You can change the following properties:

- **Direction,** sets the direction of the shadow, from 0-360°. The range of the possible directions shapes a circle.
- **Distance,** sets the distance between font and shadow. This is then the radius of the circle with possible shadow directions.
- **Z-offset,** sets the displacement of the shadow on the z-axis.
- **Color and Transparency,** you can set the color and the alpha value of the drop shadow.
- **Soft Shadow,** allows you to generate soft shadows in four different blurring stages. Click the Soft Shadow button and select the wanted blurring.

Clock



In the clock menu you insert and edit the display of a clock. You activate the clock by clicking the button labelled "Inactive". After activating a clock in the text editor you can change the following properties for the clock:

- **Inactive:** This button activates the clock. The label of the button then changes to "active". A clock symbol is visible in the text field and the digits of the clock are shown in the render area.
- **Clock:** You can have up to 9 separate clocks running individually. This gives you the possibility to display different time zones for instance. Clocks are global. If you create multiple displays of clock 1 for instance, and you stop one of them, all clocks with id 1 will stop.
- **Time,** allows you to set the starting time.
- **Limit,** allows you to set the maximum time limit.
- **Start, Stop and Cont,** buttons start, stops and continues clock
- **Format,** allows you to set the format of the clock display:
 - "h" stands for hours
 - "m" stands for minutes
 - "s" for seconds
 - "d" for digits of a second. One "d" will show tenth, two hundreds and three thousands of a second.
 - User Defined, where you can arrange it as you like and put in any separation character.
- **Up Down,** sets if the clock should count upwards or downwards.
- **Fixed,** sets a fixed text box. If your clock changes the kerning while counting, you can activate the "Fixed" button to use a fixed Kerning for each letter - The problem with changeable kerning can appear if you use a proportional font.
- **Normal,** shows a normal main clock view.
- **Count,** allows you to create a "time count" version of a standard clock, that counts a defined time lap in relation to one of the running clocks. If you for instance have created clock and it's running, you can create another container with the same clock, change its state to "Count" and define a time range. When you press the start button under the "count" section, the "count-clock" will start counting the time defined. It is related to the "main" clock, so if you stop the main clock the "count" clock will also stop. A typical use for this function is for counting penalty times in sport graphics. In a sport game running on "efficient time" a penalty count down should stop when the main game time stops.
 - **no sync,** with this option enabled, the count clock will run with no second synchronization to the main clock. It is almost impossible to start the count clock exact at a whole second so it runs synchronous with the main clock, the seconds run unevenly. This will in many cases not create the view one wants.
 - **sync sec,** when this option is selected, the count clock will synchronize its seconds view with the main clock.
 - **loop,** with this option set, the count clock will loop on the defined count range.
 - **Start, Stop, Cont and Init,** starts, stops, continues and resets the count clock.
- **Offset.** By clicking the "Offset" button, a value field becomes visible. You can enter an offset value here. The value is seconds. For instance if the

value is set to 20, after pressing the start button, 20 seconds will elapse before the clock starts running.

- **Update Text** - Switches on/off update of the text. If you switch it off, the clock will continue to run but the clock's text will stop. Use this function for showing "Split times".

Text box



In this sub-menu you can enable and scale a text box. The text box can be seen if you turn on the Bounding Box 'BB' option by clicking on the "BB" button on the left button bar in the rendering window. The text box appears as a brown outline. When you enter a text, and a word is crossing the borders of the text box, the word will be moved to the next line. There is no hyphenation/word-splitting function. The text on each line automatically gets the selected justification. NB! When using a IME editor with complex script option set as the IME type, the box option does not work!.

Show icons

In the lower right corner of the text editor a button labelled "Show icons" is located. By clicking this button, a list of the symbols in the font's library is shown as icons on the right side of the editor. As an alternative to using the keyboard, you can drag and drop these symbols onto the text pad. This is a god aid when using special characters that don't exist on the keyboard.

Using the IME-based editor

For those who need to use complex Unicode fonts like Chinese and Japanese, Viz|Artist have an IME (input method editor) which allows you to generate text with these fonts. An input method editor (IME) is essentially a facility by which you can enter these complex characters while using a standard vanilla U.S. keyboard. To enable this editor you must chose the UTF-8 font code in the config section under "font codes". The text editor then gets a new button labelled "IME". Press this to initialize the editor. To make the editor work you must in addition install and import into Viz|Artist, the Unicode font: "Arial Unicode MS". You must use this font in the scenes for generating the text. In the "Regional settings" sections in windows control panel, you must also install and enable the languages you need. Enable the option "enable indicator on taskbar". The language settings are application specific so you must toggle between the languages with Alt+LeftShift when you are in Viz|Artist. The IME functionality is not similar for all languages; refer the documentation for each of the IME languages.

Working in the Rendering Area

Editing containers in the rendering window

The rendering window includes edit functions. Some of the functions have a small status window that is displayed beside the cursor, monitoring the properties, while you are editing a container.

Select and Position a container

You can pick a container by using left mouse. This is done on a per pixel basis. To indicate that the container is selected, the bounding box shows for the object, if the BB-option is enabled, and the entry for the container in the

transformation tree is colored blue. You can thereafter move it by holding down left mouse button and moving the mouse. The object stays exactly under the mouse cursor. If you hold down the Shift key as you move the object, the axes are locked, and you will only be able to move the object parallel with the X- and Y-axis.

Rotate an container

You can rotate a marked container using middle mouse button. Make sure the container is selected, click on it with the middle mouse button and keep it pressed. Move the mouse to rotate the container.

Move a container in Z

You can move a container on Z-axis by using left+middle mouse button. Make sure the container is selected, click on it first with the left button, hold it down and press the middle button. With both buttons pressed, move the mouse sideways to move the container on the Z-axis.

Copy a container

Hold down the Ctrl key before clicking on a container and dragging it to the wanted position to copy the container. If you in addition hold down the shift key, you can copy containers evenly along the X- and Y-axis.

Editing with the right mouse button menu

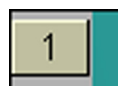
With a container selected, the right mouse button gives you a sub menu: Press the right mouse button and then select 'scale', 'rotation', 'center' or 'animation'.

- **"Scale"** displays scaling handles at the bounding box. Pick any of the yellow handles and drag with the left mouse button to scale the object.
- **"Rotation"** displays rotation handles on the axis. Pick it with the left mouse button and drag the mouse left or right to rotate the object around an axis.
- **"Center"** displays handles to move the object centre (these handles look the same as the rotation handles). Dragging the centre handles allows you to shift the axis centre of the object e.g. to have it rotate around another centre.
- **"Animation"** shows the animation path and the position key frames of the container. This element is explained in the animation section, see ["Shader Function plug-ins" on page 155](#).

Additional functions connected to the output window

The left side button bar

On the left side of the window there are several buttons:



This is the camera selection button. The number on the label shows the current camera. If you click it you get a menu:

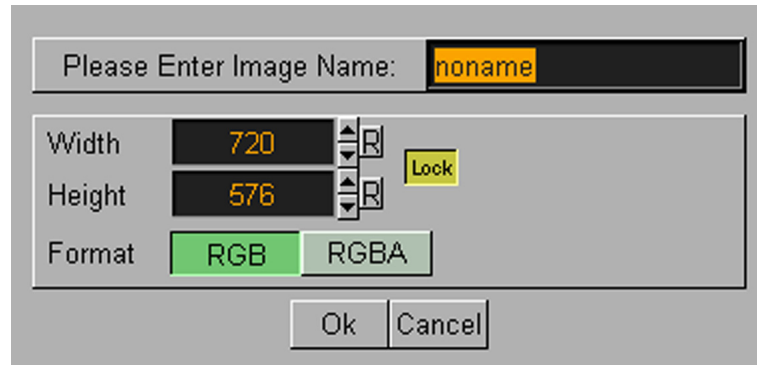
1	1	3	5	7	9	11	13	15	Front	Left	Top
	2	4	6	8	10	12	14	16	Back	Right	Bottom

1-16 are the different cameras. Click to select camera.

Front, Back, Left, Right, Top and Bottom are six predefined orthogonal camera views. In these views you can edit light and camera positions directly.

Snap

Snap opens a popup window where you can take a snapshot of the current state in the rendering window.



Enter a name, change width and height if desired, choose format with or without alpha channel and click ok to take snapshot.

TA

TA displays the title area in the render window.

SA

SA displays the safe area in the render window.

Key

Key gives you a view of what the Key output will look like. You cannot do editing (picking...) in the Key view.

W

W renders the scene in wire frame mode.

BB

BB switches on and off the bounding box visualization of the container currently selected.

P

P opens the performance meter. It gives you an idea of the performance (frames per second) this scene is rendering at. The performance meter contains a bar with different colors and some performance data.



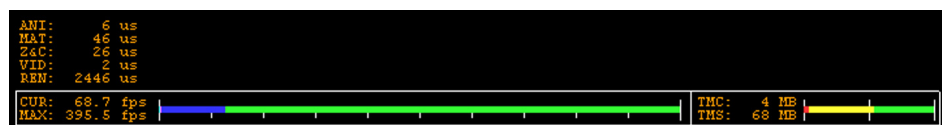
- The **CUR** value tells you how many frames per second this scene will render at in on air mode. It should be above 50 (PAL) or 60 (NTSC) according to which rate has been specified in the configuration editor.

- The **MAX** value tells you how many frames per second the scene can render at without waiting for vertical retrace. The higher the second number the more performance is left. If the second number goes down below 50 or 60 the scene is not rendering in real-time.

To the right are two values that give you information about texture memory.

- The **TMC** value gives you the texture memory in use by all scenes loaded in memory.
- The **TMS** value gives you the total amount of texture memory allocated by the system.

If you click on the “P” button with the centre mouse button some additional advanced performance indicators appear. They show how many microseconds each rendering step takes. They are also linked to the bars with different colors:

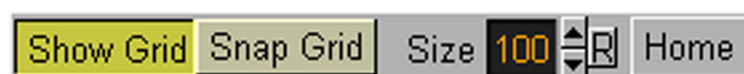


- The **ANI** value (Yellow bar), calculates all active directors and animation channels.
- The **MAT** value (Cyan bar), transforms each container in the scene into world coordinate space.
- The **Z&C** value (Pink bar), Z-Sort and Culling: Sorts all containers for correct transparency drawing and determines if containers are visible in the current camera view.
- The **VID** value (Red bar) Shows how long video input (live video texture) and video output take. Deinterlaced video inputs take longer than progressive and interlaced. The only way to improve this value is to use a faster system.
- The **REN** value (Blue bar), shows how long it takes to render all objects on the screen. A faster graphics card will improve this value.

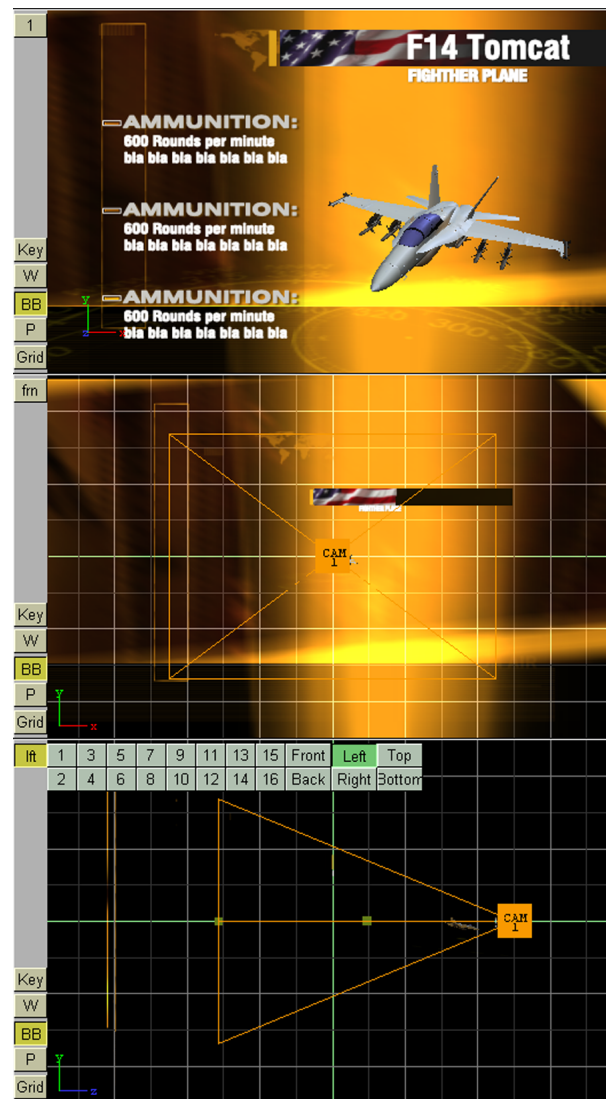
Grid

The grid button shows the grid editor. The grid is shown in all orthogonal views. You may switch it on or off and change its size. 100 mean 100 centimetres. The perspective camera is 5 meters away from the zero point.

Snap Grid enables a “Snap to Grid” function. With that enabled, when you move an object in the render area it will snap to the nearest grid when you release the mouse button.



Using the side-views



By clicking the Views button in the main menu three extra rendering windows open on the left side of the screen. These are a good aid for getting an overview of the objects in the scene and visualize their position in relation to lights and cameras. They have in principle the same functionality as the main rendering window. You can change camera view, position objects, lights, cameras and more. If you set them to display one of the six orthogonal side views there are some special short keys to use for navigation:

- **Pan** by clicking the left mouse into the background. If the background is obscured press the c-key in addition to moving the mouse with the left button pressed, to pan.
- **Zoom in** by holding down the z-key and draw a rectangle with the left mouse button pressed. Pressing the z-key multiple times zoom in a stationary position.
- **Zoom out** by holding down the x-key and draw a rectangle with the left mouse button pressed. Pressing the x-key multiple times zoom out in a stationary position.

In the six predefined side views, the currently selected camera, and in addition all cameras that have been given the status "vis" or "inf", will appear as symbols. By clicking on them in the side views, their focus will be set in the main rendering window and handles for editing position and direction will be visible. Use the mouse and drag the handles to change position and direction.

The same is the case with lights. The one currently selected and those who have a "vis" status, will be visible as symbols and position and direction can be edited with the mouse.

Using the main function plugins

Using the Key function

Note!

If you have a graphics card that support shader technology, you should use the **Auto Key** function instead of the key function: See, "[Scene setup Background page](#)" on page 172 in the

The key editor opens when you click on a key icon in a container. You can change the key-options here. If you want to look at the key signal, click on the Key button on the left side of the rendering window. The parameters in the key editor are as following:

- **Draw RGB**, disable this if you don't want the graphical object to be shown, just the key signal.
- **Draw Key**, enables/disables the key signal.
- **Automatic Alpha**, if you disable the automatic alpha option, you can set the value manually. This is normally done to obtain some level of transparency so the e.g. video background is visible through the key object.
- **Alpha as key only**, takes the alpha value of the object and renders the key with this alpha instead of the alpha value of the key function attached. The object itself is rendered opaque. This is usually used to render transparent objects for keyed graphics. The "color mixing" of the object in the scene and the e.g. video background is done by the external keyer so the object must be rendered non transparent.
- **Renderoptions**, this option sets the mode in which the alpha values of the key element you are rendering should be mixed with the alpha values already existing in the frame buffer. Be aware that with the subtractive and the rev-subtractive mode the order in which the objects are being rendered then is crucial. So make sure the z-sort is correct. The mixing is being performed on a per pixel basis. (in the formulas the range of the alpha values is from 0-1, instead of 0-100 as in the value field. If we say A is the current source alpha value being rendered now and FA is the alpha of the frame buffer contents (target), the different modes can be explained as following:
 - **Additive**, symbolically: $A + FA$. The source alpha gets added to the target alpha. As the formula shows it is an addition, so the end result will always be a higher alpha value than the initial frame content. If the frame alpha has a high value, you might experience that the addition gives a total value above 1. This will be clamped at 1 which is opaque.
 - **Blended**, symbolically: $A + (1-A) * FA$. The new alpha value gets created as a weighted average of the source and the target. If we as an example say that the key being rendered has alpha 1 and the frame alpha has 0.5, the formula will be like this: $1 + (1-0.5) * 0.5 = 0.75$.
 - **Subtractive**, symbolically: $FA - A$. The new alpha value is the result of the incoming alpha being subtracted from the alpha in the frame buffer and the result is written back into the frame buffer.
 - **Rev-subtractive**, symbolically: $A - FA$. This is a reversed version of subtractive. The alpha in the frame buffer gets subtracted from the incoming alpha and the result is written back into the frame buffer.

Using the Expert Function

The Expert Editor opens when you click on an Expert icon in a container. The expert Function contains some advanced functions and settings. By dragging an Expert Function to a container you can set the following parameters:

- **Draw mode**; you can chose to have the container drawn the following ways:
 - Normal
 - Wireframe
 - Outline
- **Width**; sets the width of the wires if “wireframe” or “outline” draw mode is selected.
- **Backface**; by enabling this option, an image or a 2D font will be drawn with a backface. This will of course take more performance to render.
- **Two-sided lighting**, with this parameter two-sided lighting can be enabled for a single container instead for a whole scene.
- **Z-buffer Draw**, allows you to disable the OpenGL z-buffer draw. This is in some cases the last possibility for solving some z-sort problems. The technical explanation for this is very complex and will not be described here.
- **Line Antialiasing**: 2D objects on machines that don't have antialiasing with multisampling, an antialiased outline are rendered instead. This extra outline can be removed by disabling this option.
- **Mirror**, enables you to mirror an image/texture or a 2D font over the X-, Y- and Z-axis.
- **Renderoptions**:
 - **Additive**, symbolically: $C*A + FC$. The source color gets added to the target color. The amount of color that is added depends, as we see from the formula, on the alpha value. However, it is always an addition, so the end result will always be a lighter color than the initial frame content. If the frame color has high values on all three color channels (RGB), you might experience that the addition of the new color takes all channels to values above 255 (saturation). The values will be clamped at 255, which is white.
 - **Blended**, symbolically: $C*A+(1-A)*FC$. The new color value gets created as a weighted average of the source and the target. The weight factor is the alpha value of the rendered color. That means that if the incoming color has a very low alpha value, its influence on the new color will be small, and conversely, if the new color has a high alpha value, its influence on the new color will be bigger.
 - **Subtractive**, symbolically: $FC-C*A$. The new color is the result of the incoming color being subtracted from the color in the frame buffer and the result is written back into the frame buffer. The alpha value of the incoming color, decides how much who gets subtracted. If the incoming color has high values on all three color channels (RGB), you might experience that the subtraction of the new color takes all channel values to 0, and the result is a black color.
 - **Rev-subtractive**, symbolically: $C*A-FC$. This is a reversed version of subtractive. The color in the frame buffer gets subtracted from the incoming color and the result is written back into the frame buffer.
 - **Multiply**, symbolically: $CA*FC$. The new color and alpha get multiplied with the existing values in the frame buffer. The formula presupposes that the colors and alpha re described as values between 0 and 1. A color rendered with multiply will always result in a darker color than both the color being rendered and the color in the frame buffer.
- **Shade Model**
 - **Flat** - Shows the single polygons the texture is made of.
 - **Smooth** - Calculates a smooth texture.
- **Transparency Model** - On graphic boards that support shaders. Different ways to render transparency can be used.
 - **Blend** - Is the standard option
 - **Mask** - Uses a raster
 - **Fastest** - The model that takes the least time to render
 - **High Quality** - gives the best visual result

Using Mask Function

Mask and Source layers.

With the mask function you can make one object acting as a mask for another. The mask object is called the source and the object being affected by the source mask, is called the target. An object being affected by a mask becomes transparent at the area where the mask covers it. There is one editor for the mask source and one for the mask target, where you can define by which layer the object is affected. To make an object a mask source or target, drag the mask source function or the mask target function onto the container. You can use up to 8 different target layers and source layers for one object. This means that targets with the layer 1 set will be affected by all masks which have the corresponded layer selected. If a mask has the layers 1, 2 and 3 selected it will affect all targets with 1, 2 or 3 selected.

Inverted mask

In the setup menu under the heading "background", you can define if a mask layer is to be drawn inverted. Normally the mask cuts a hole in the mask target where it covers it. If you select a layer to be drawn inverted the opposite situation is the case. Only where the mask covers the mask target will the target be visible.

Using the Window Mask function

The Window Mask defines a 2D rectangular mask for viz containers. When you place a Window Mask function on a sub-tree, this sub-tree will be rendered only inside the area covered by the rectangular mask. Although you can achieve the same results using a normal mask and mask-target functions, Window Mask has very significant performance advantage. The pixels outside the rectangular mask are not processed by the graphics hardware (unlike normal masks). In addition, normal masks have a very significant performance penalty on some graphics cards. Window Mask has a no significant performance penalty.

Using Extrusion function

Extrusion, general description

You can use the extrusion function onto a container using 2-D primitive or a font. An extrusion can be explained simply as extruding or sweeping a 2D shape through space. A "copy" of the original primitive or font is made at a point with some Z-value offset from the main figure, and the edges of the original and the "copy" are interpolated. The result is a 3D image of the 2D shape.

Extrusion Editor parameters

- **Bevel type.** The buttons marked with different bevel types, sets the type of bevel to use on the figure.
- **Extrusion depth,** sets the extrusions depth on the Z-axis.
- **Bevel size,** sets the size of the bevel. This must be set to some value to enable the selection of bevel type.
- **Shading angle,** sets the angle of the shading function. To "erase out" unevenness, try increasing the shading angle. On shading, see "[Shading](#)" on page 55
- **Bevel detail,** sets the degree of detail of the bevel. The lower the value is set the more detailed the bevel is constructed. The more detailed the bevel is the smaller tilings it's constructed from.
- **Backface,** you can enable or disable visualization of the backface.

Using the Shadow function

The shadow function consists of two function parts, the Shadow Caster- and the Shadow receiver function. To create a shadow element in a scene, two containers minimum must be used. The Caster function must be attached on

one container and the receiver function on the other. The container acting as the caster must be positioned between the receiver container and a light source.

- The caster function, see [“Using the Shadow Caster Function” on page 68](#)
- The receiver function, see [“Using the Shadow receiver function” on page 68](#)

Using the Shadow Caster Function

This editor opens when you click on a Shadow Caster function in a container. It allows you to set some parameters on how the caster object casts shadow on receiver objects. The parameters are as follows:

- **Lights**, allows you to select which light sources that are to shine on the caster object and thereby producing the shadow/shadows.
- **Alpha**, sets the alpha value and hence the transparency of the shadow being cast on the receiver objects.
- **Color**, allows you to define the shadows color.
- **Auto Fading**: With this option enabled, if you fade out a caster object, either by using the alpha function or by adding a material with a low alpha value, the shadows created by the caster will fade correspondingly. The option is enabled by default, since it looks right that the shadow fades out when the caster object fades out.
- **Layer**: Here you select by which layers the caster will cast shadow. You can select up to 8 layers. Each layer corresponds to the layer selected in the Receiver objects. If you want all shadow casters to cast shadow on all receivers, you can set all layers on all casters and receivers to 1. If you want one caster to create shadow on some receivers and another caster to create shadow on different receivers, you select different layers for the sets of casters and corresponding receivers. Casters and receivers can have multiple layers enabled at the same time so you can set up a great number of combinations of caster- and receiver combinations.

Using the Shadow receiver function

This editor opens when you click on a Shadow Receiver function in a container. It allows you to set some parameters on how the receiver object receives shadows produced by a corresponding shadow caster object. Objects that can function as a shadow receiver should be a built in 2-D primitive or a font. An imported 2-D will not work. The parameters are as follows:

- **Alpha**, sets the alpha value and hence the transparency of the shadow.
- **Clipping**: With Clipping enabled, the shadow will be cut on the border of the object. With clipping disabled, the shadow will be rendered onto the background outside the object, which normally looks wrong. Rendering with the clipping options set takes more resources than without. If you for instance have an infinite floor or a room with walls, you can switch of clipping without seeing the “false” shadows, and thereby save some rendering performance.
- **Culling**: With culling enabled, the program will decide in advance of the rendering process if the shadow will be visible or if it will be hidden by other objects. If that is the case it will skip it in the render process. If you know for sure that the shadow for an object in a scene will be rendered visible for the whole object, you can disable culling to save some rendering time, but the effect is marginal.
- **Show receiver**, hides or shows the receiver object. If it is hidden, the shadow will still be visible.
- **Draw Key** - If the receiver object is hidden you can set this option to still have the key draws.
- **Key Value** - Sets the alpha for the receiver key.
- **Layer**, Here you select by which layers the receiver will receive shadow. You can select up to 8 layers. Each layer corresponds to the layer selected in the caster objects. If you want all shadow casters to cast shadow on all receivers, you can set all layers on all casters and receivers to 1. If you want one caster to create shadow on some receivers and another caster to

create shadow on different receivers, you select different layers for the sets of casters and corresponding receivers. Casters and receivers can have multiple layers enabled at the same time so you can set up a great number of combinations of caster- and receiver combinations.

Using the Projector function

General description

The projector function consists of two parts, the projector and the projector target. The container/containers with the projector function attached will create a planar projection of their texture onto containers with the projector receiver function attached. It works in principle in a way similar to a Dias projector. For objects that don't have any texture coordinates, the projector function is a possibility to apply a texture to such objects. It gives you also the possibility to have two textures on a single object, the object's own texture and the projected texture. The projector and the projector receiver can be set up to react on different layers similar to shadows and masks. The function does not work on text.

Projector parameters

- **fovy**, sets the field of view for the projector container. The higher the fovy is set the larger the projected texture on the receiver will be. You can picture it as the zoom function of a dias projector.
- **Show**, allows you to hide/show the projector object.
- **Layer** (both projector and projector receiver), let you select by which layers the projector and receiver are to perform the function. You can use up to 8 different projector layers and source layers for one object. This means that targets with the layer 1 set will be affected by all projectors which have the corresponded layer selected. If a projector has the layers 1, 2 and 3 selected it will affect all targets with 1, 2 or 3 selected.

Use of the projector function

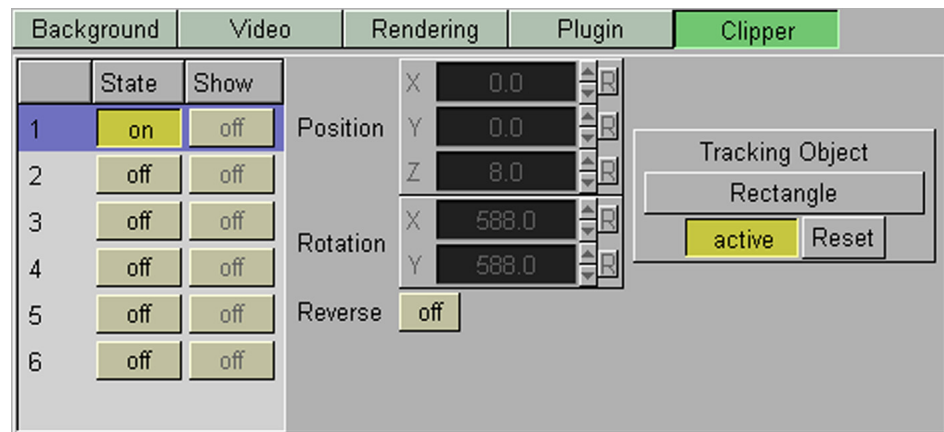
The projector function gives you many creative possibilities:

- It is a possible solution if you want two textures on one container. This means that the projector target objects can have their own texture in addition to the one being projected on them by the projector object. To make them blend together you must add the expert and the alpha function onto the projector container. In the expert editor set the rendering mode to "add" and in the alpha editor set the alpha to some level of transparency. Then the projected texture will blend with the projector target's own texture.
- By projecting a greyscale image of a circle or similar, you can simulate a spotlight. Also here is the use of the expert function rendering modes essential.

Using the Clipper function



The clipper function is an alternative to the mask function. Objects with the clipper function attached that are placed behind the clipper plane will be masked/clipped. If an object is only partially behind a clipper plane, only the part of the object that is behind the plane will be clipped out, since the clipping is done in through 3D space. The function uses the OpenGL clipping planes. There are 6 of them available. You define the clipper planes in a editor in the setup menu:



You can either define the clipper plane with position and rotation, or you can let the clipper plane track a 2D object (e.g. rectangle). To use a tracking object, drag the object's container from the tree and onto the tracking object dropzone. The advantage of using a tracking object is that you can animate it! The Reverse button reverses the clipping plane front/back.

On the containers that you want to be affected by the clipping plane, you must add the clipper plug-in function and thereafter in the editor specify which of the clipping plane that is to mask out/clip the object. It is possible to let an object be affected by multiple clipping planes.



The clipper plug-in can replace mask in many situations. Use it if you can since it has some advantages compared to normal mask:

1. Clipping is done in true 3D space, while masking is done in 2D (although the 2D mask is created by a 3D object).
2. There is no performance hit. In fact there is even a performance gain if parts of the object are clipped. Regular masks have a quite high performance cost on a PC platform.

Lights and cameras

Using Light Editor

Editing the Light settings in a scene

The light editor opens when you click the Light button on the main menu bar. It allows you to change the light settings in the scene. A scene can have up to 8 simultaneously active light sources. The lights can be defined to be different light types, and you can edit their position, distance, angle and color. The paragraphs below explain the basic editing possibilities.

Switch lights on and off

The buttons in the "State" column allows you to define if a light source is to be active or not. Press the button the change its state from on to off or vice versa. The handles belonging to the light source currently selected will be shown as long as the light editor is open. By clicking on the button in the "Vis" column, and set it to "on", the light handle stays visible and accessible for editing even if the light editor is not open. As long as the bounding box option is enabled, the light source that is selected for editing will always be visible even if the vis option is off.

Different type of lights

By default when a new scene is created all lights are infinite. Infinite lights have adjustable light direction. You can change the light type to:

- **Local light**, which is a positional light. It is near or within the scene and the direction of its rays is taken into account in lighting calculations. Local lights have a greater performance cost than directional lights due to these additional calculations. A real life equivalent to a local light source would be a light bulb.
- **Infinite light**, which is a directional light source. It is considered to be an infinite distance away from the objects in the scene. Because of the distance, its rays of light are considered parallel by the time they reach the object. A real life equivalent to an infinite light source would be sunlight.
- **Spot light**, which emits a cone of light. The only objects illuminated are those within this cone

Position and direction of lights

- **Position and direction:** With a Local light source you can only change position, since it has no directional qualities. With infinite- and spot light sources you can change position and direction of light.
- **Attenuation:** A spot light can in addition have its attenuation defined. This is an exponent that describes the concentration of the light within the light cone of the spot. When set to zero the whole light cone has the same intensity. If set greater than zero the intensity decreases away from the centre. This parameter becomes visible when "spot" is selected.

Editing techniques for setting position and direction

Each light source's position and direction can be set by altering the corresponding x-,y- and z-values in the editor when the light source is selected. It can also be done directly in the rendering window by repositioning of the light handles. To do this, the light must be selected in the editor or the "Vis" option must be enabled for that light source. In the rendering window, a local light are visualized as a single point, Infinite and spot light, are visualized as a line with three small squares, which are the handles. The spot light have in addition a "wireframe box" that shows the attenuation or in other words, the scope of the light. The squares at the ends of the light source slightly differ in size and color (when light source is not selected). The bigger one is the light source and the smaller is the handle that sets the direction and distance of the light. The middle square is a handle that can be used to move the whole light source without altering the proportionality of position and direction.

Track object with light source

If you drag an object out of the Transformation Tree into the "Tracking Object" areas the position/direction of the light source will follow this object. This is useful when you animate an object. If the light position or direction is tracked, you can deactivate it using by clicking the active-button, changing its state to "inactive", or remove the tracking object by clicking the reset button.

Animate Lights

You can also animate the light in the same way as objects, by creating keyframes. Select the light source, move it to the new position or/and set a new direction and add a new keyframe by pressing the update button in the rendering window. The attenuation value can be animated for a spot light.

Using the Camera Editor

Editing the Camera settings in a scene

The camera editor opens when you click on the camera button in the main menu. It consists of two windows. The left window contains a list of all 16 virtual cameras and the right window holds the parameters for the camera cur-

rently selected. The camera you select in the camera list will automatically be set as viewing perspective in the rendering window. In the camera list by each camera there is a button under the column heading "Vis". Default value is "off". By clicking the button you can change its label to "vis" or "inf". If the one of these statuses is set, the camera will appear as a symbol in the six predefined side views. The camera symbol has handles that you can use to alter camera position and direction. The "inf" option differs from "vis" with its infinite lines on the camera symbol, that visualizes the camera view. This is a good aid for visualizing the scope of a camera that is located far away.

Selecting camera

The camera view of the main rendering window can be set either by selecting a camera in the camera editor or by selecting from the camera button in the side menu bar of the rendering window, see ["The left side button bar" on page 61](#)

Camera views

The basic view of a virtual camera is a normal depth perspective. Cameras can also be set to display orthogonal views, if one for some reason wants to have a view without perspective. Without perspective objects will not change their size when being moved towards or away from the camera, and in some cases this might be a desired effect. All parameters except X- and Y-position will then be disabled.

Side views

The six predefined side-views are orthogonal views, (Front, back, right, etc.). They are especially useful when you want to arrange and fine tune positions of objects, lights and cameras. In these views there are some special shortcuts accessible:

- **Pan** by clicking the left mouse into the background. If the background is obscured press the c-key in addition to moving the mouse with the left button pressed, to pan.
- **Zoom in** by holding down the z-key and draw a rectangle with the left mouse button pressed. Pressing the z-key multiple times zoom in a stationary position.
- **Zoom out** by holding down the x-key and draw a rectangle with the left mouse button pressed. Pressing the x-key multiple times zoom out in a stationary position.

Position, Zoom, Pan, Tilt and Focus.

A virtual camera is able to perform all movement and operations of a normal studio camera. It can:

- Be positioned on the three axis, X, Y and Z
- Be panned, tilted and twisted
- Lens operations like zoom, focal plane and shift of lens centre can be performed.
- A relative position to a real camera can be defined.

Shortcuts for orienting the camera

The table below shows a set of key+mouse combinations that allows you to move the camera currently selected in different ways. Select the combination you want an move the mouse sideways to perform the wanted action. The shortcuts work both in the main rendering window and in the three additional side views.

Function	Key+Mouse combination
Position X	P + left mouse button

Function	Key+Mouse combination
Position Y	P + middle mouse button
Position X	P + right mouse button
Position X/Y	P + left and middle mouse button
Position X/Z	P + left and right mouse button
Pan	I + left mouse button
Tilt	I + middle mouse button
Twist	I + right mouse button
Pan/tilt	I + left and middle mouse button
Zoom	U + left mouse button
Orbit X	O + middle mouse button
Orbit Y	O + left mouse button
Close in/out	O + right mouse button
Orient against selected object	Press the T key

Camera shortcuts to the six orthogonal views

Use the numeric key pad to toggle between the six orthogonal view. They work both in the main rendering window and in the three side views.

Key	View
0	Back
7	Front
2	Bottom
8	Top
4	Left
6	Right

Track an object with the camera

A virtual camera can be set up to track objects both on position and direction. Drag the desired container onto the dropzone of either position or direction to enable the tracking. The name of the selected container is then written on the dropzone button. If you click on it, the transformation editor for the object is opened and the container is selected in the tree. You can make the tracking inactive by pressing the button labelled "active", it then changes its label to "inactive". Click on it again to reactivate the tracking. To disable the tracking function, click the "reset" button.

Receive tracking data from a real camera

On virtual sets, as the real cameras moves the virtual camera should move in the same manner. To enable a virtual camera to receive positioning data from a real camera, click on the button in the Control column and set it to

“Remote”. On how to set up and define links to external cameras, see the documentation for the virtual set option.

Copy properties from one camera to another

If you want two cameras to have equal properties, in the camera list you can drag the name section of one camera onto another. The target camera will then inherit all properties belonging to the source camera.

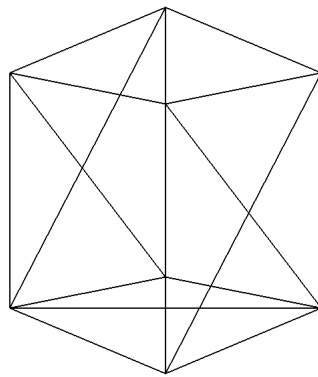
Animate camera settings.

All parameters for the cameras can be animated in the normal way, by setting a value, pressing update, set a new value and pressing update again. Animation can also easily be done in one of the six orthogonal views. There the cameras appear as symbols and you can position them with the mouse, thereby creating an animation sequence. More advanced camera animations are normally done with animation of tracking objects.

Built in objects

Recurring parameters

Tessellation



All built in primitives in Viz|Artist are polygons made up by smaller polygon elements. It looks like a mosaic. By using many small polygons instead of fewer and bigger polygons, a more precise construction of the object is made. The tessellation parameter that we find on almost every Viz|Artist primitive sets the degree of tessellation on the object, meaning the size and number of polygons it should be built up of. The higher the tessellation value is set, the more fragmented the mosaic pattern of tiling pieces are. A good way to visualize this is to look at an object in wireframe, and then alter the tessellation parameter. You then see how the web of small pieces gets more fragmented as the value increases. An important thing to remember is that the higher the tessellation is set the more polygon vertices are created and this of course influence the rendering performance. The calculation of light on an object need some level of tessellation. Since light is calculated at an objects vertices, the higher the tessellation is set, the better the rendering of light will be. You can easily try this by creating a floor with a rectangle and change its tessellation. The light on the object will change as the tessellation changes.

Use LOD (level of detail)

This parameter is present on almost every Viz|Artist primitive. It's enabled by default and sets a dynamic detail function on the object. It takes advantage of the fact that small objects, scaled small or placed far away from the camera, do not need to be as detailed as bigger objects of objects close to the camera.

It reduces dynamically the detail level on smaller and distant objects, by setting down the tessellation value. This is done in a way that in most cases, not influence the realism of the objects presence. Through this much rendering performance is saved, by not having to render distant objects with a high detail level where it is not needed. You can see this effect easily:

- Take a primitive e.g. the cone and set it in wireframe mode.
- Set the tessellation at a high level.
- Increase the distance between the object and the camera slowly. At some point you will see that the degree of tessellation is being reduced.

Polygon



General description

The Polygon built in object enables you to create a user defined polygon. The polygon is a 2D object. The extrusion function can be applied to it to make into a 3D object. When you drag a polygon object onto the transformation tree, the polygon drawing pad, opens in the upper right section. You create a polygon by adding intersection points. Left click in the drawing pad, move the mouse to a new position and click again. An intersection point is created each time you click, and a line is drawn between the point and the last one created. The polygon border line is built up by locally managed spline curves.

Maneuver on the drawing pad

- To alter the view offset and move around on the pad, press the middle button and move the mouse.
- To return to start position on the pad, press the HOME-key.
- To zoom in, press the Z-key
- To zoom out, press the X-key
- To set zoom back to 1, press CTRL+HOME
- To set zoom factor from 1-9, press CTRL+1-9
- To set zoom factor from 0.1-0,9 press SHIFT+1-9

Select existing elements for editing

When you want to edit an existing element of the polygon, you must point exactly at it with the mouse pointer. To help you see when an existing element is selected for editing, the color of a line section selected, changes too a light green as you move the mouse pointer above it. When you hold the pointer over an intersection point, a text box appears which shows the number and position of the point. To select the point, left click once. The color of the point changes from yellow to blue. To select more points at the same time, with an intersection point selected, right click and choose spline from the menu, and thereafter the sub element "select". You can choose between the following options:

- **All**, selects all spline sections of the polygon. (Short key CTRL+S)
- **To Begin**, selects all spline sections from the point selected and to the beginning of the polygon.
- **To End**, selects all spline sections from the point selected to the end of the polygon.
- **Invert**, turns around the selection already made. If all is selected, invert will change it to none selected and conversely if "To begin" have been selected, it will transform to "To End" after invert has been selected.

Altering the shape of the polygon

You can move an existing intersection point by dragging it along with the left mouse button pressed. At each point spline handles is attached. They have an

equal functionality as the spline handles on the animation curves in the rendering window. (The spline handles set the tangent vector). When you, with the left mouse button pressed, alter the position of the handles, the local section of the polygon, connected with the spline handles, changes its form.

Add new intersection points within spline.

You can add new intersection points by left clicking on a section of the polygon border. A new point with spline handles attached will then be created.

Close/open the spline

To close the polygon between the last and the first point, press C or right click, and from the menu choose properties>spline>close. Doing one of the two, once more, opens the spline again.

Delete an intersection point or a border section

To delete an intersection point, select it by holding the mouse pointer above it, right click and select "cut" from the appearing menu. By pressing the delete button on the keyboard, the intersection points get deleted one by one, starting on the highest number. To delete the whole spline object, press CTRL+Delete.

Copy/cut and paste intersection points

To copy or cut points, select them, right click and select copy or cut. To paste, right click and select paste. To copy you can also use the short key CTRL+C. You can in addition to pasting the points as they are, paste the points you have on the clipboard in the following ways:

- **Reversed**, this option will paste the points in a reversed order. This is the same as mirroring over X and Y together.
 - **Mirrored X**. This option will paste the points mirrored along the X-axis.
 - **Mirrored Y**. This option will paste the points mirrored along the Y-axis
- Take notice that it is just the points being mirrored or reversed, not the spline handles. This will in many cases make the pasted polygon look quite different from the original, unless all points are linear.

Spline handle modes

By default, the spline handles to the right and left of each intersection point are set up to be looked. This means that if you alter the position of one handle, the other changes correspondingly. You can alter the mode for the spline handles by right clicking, in the menu choose properties>handle mode. The modes are:

- **Looked**, the handles are looked so they always create a straight line. If you move the left one down on the y-axis the right one moves up etc.
- **Single**, the handles can be adjusted individually.
- **Mirrored**, the handles get mirrored through an invisible line drawn 90 degrees on both handles as they were when the intersection point was created.

The handle Mode menu has two more options:

- **Auto recalc**, with this option set, the spline handles at each point is automatically recalculated in relation to the point before and after when changes have been made. If it is disabled, all points will be created without handles and you must enable them manually.
- **Show always**, if this option is deselected, the handles only shows for the intersection points currently selected.

Set or hide anchor

An anchor creates a point around which the spline rotates; a centre point so to say. To set an anchor right click at the wanted point and select anchor>set.

You can move the anchor afterwards by clicking on it with the left mouse button and drag it to a new location. To hide an anchor, right click and select anchor>hide.

Split spline sections

Splitting a spline section can be done in three ways:

1. Select the section, right click, and from the menu select spline>split.
2. Select the section and press S.
3. Press SHIFT+S, the pointer symbol changes to a pair of scissors, click on the desired line section.

After you have split the section, the numbering of the points changes.

Flip spline sections

You can flip selected or all spline sections over the X- or Y-axis. Right click and select spline>flip>All X, All Y, Selected X or Selected Y. What the flip function does is to switch the X or Y values between the points in an order where the first point switch value with the last point, the second point switch value with the last but one, and so on. To make a visualization of the effect, make a upward straight line with five points and flip all points over the x-axis. The result should be a downward line.

Mirror spline sections

By right clicking in at the pad, choosing spline>mirror, you can mirror selected points or all points along the X- or Y axis. The points get mirrored through the average value of the first and the last point

Reset spline handles to original position

To reset changes made to spline handles, select the handles you want to reset, right click, choose spline>point>reset handles from the menu.

Move, Scale and rotate.

If you press the space bar on the keyboard, a bounding box is shown around the polygon. This enables some new features:

- **Scale** the polygon by dragging the scale handles on the bounding box. The mouse cursor changes its stage to a two pointed arrow when the scaling mode is enabled. The scaling is preformed in a single mode, so if you only scale one of the two axes, the polygon will be stretched or compressed.
- **Rotate** the polygon by placing the mouse cursor outside one of the bounding box corners. The mouse cursor then changes to a bend two pointed arrow to indicate that the rotate mode is enabled. Left click and hold the button down as you move the mouse to rotate the polygon.
- **Position** the polygon by pointing at the cross in the centre of the bounding box, press left mouse button and move the mouse.

Use linear lines between points.

If you want some or all lines to be drawn without splines, select the points you want to alter, right click, choose spline>point>sharp edge from the menu. To have all points drawn without splines, press the L-key.

Add a new polygon

When you press the Insert key a small +-sign is shown next to the cursor. If you then click at the drawing pad, the first point of a new polygon is created.

Add a new polygon at the top of an existing

If you create a new polygon or you paste a copied polygon element, onto an existing polygon, a new separate polygon gets created on top of it. This will act as a mask on the existing one, and make it transparent where it covers the other.

Import Adobe Illustrator™ polygons

Adobe Illustrator files up to version 7.0 can be imported in the polygon editor. Add a polygon to your scene, open the editor and press import. Only polygon outlines are supported.

Short keys overview

- **HOME**, takes you to the start position in the drawing pad (0/0).
- **CTRL+HOME**, sets zoom to 1.
- **ESCAPE**, switches between splines with handles and no handles.
- **DELETE**, deletes the selected points. With no points selected, it deletes the point with the highest number.
- **CTRL+DELETE**, deletes the complete spline.
- **SPACE**, turns the bounding box on/off.
- **CTRL+1 – CTRL+9**, set the zoom factor, from 1-9.
- **CTRL+SHIFT+1 – 9**, set the zoom factor, from 0.1-0.9.
- **A**, enables/disables mode where control handles only are visible on points currently selected.
- **CTRL+A**, sets/deletes an anchor point at the mouse position.
- **B**, changes color scheme between white and black.
- **C**, closes/opens the spline/polygon.
- **CTRL+C**, copies points onto the clipboard.
- **I**, enables/disables the spline-info.
- **L**, switches between use of spline handles at every point or without handles (linear lines between points).
- **Insert**, displays a + sign. With that visible a new polygon will be created when you left click. Press again on the insert key to remove the + sign if you do not want to create a new polygon.
- **S**, cuts the spline on actual mouse position
- **SHIFT+S**, changes mode into "cutting-mode".
- **CTRL+S**, select all points on actual spline.
- **CTRL+V**, paste the points from clipboard
- **X**, zoom out.
- **Z**, zoom in.
- **CTRL+Z**, sets zoom factor 1
- **CTRL+X**, cuts selected points and set them onto the clipboard

Alpha Map



General description

The alpha map is used for objects who can't use an alpha image as an alpha channel directly. An image can have another image as alpha channel but a video source can not. More on images, see ["Use of two images in one container" on page 57](#). So if you want a video source to have an overlying alpha mask, you can use the alpha map. Drag the map onto the tree, drag the wanted alpha image onto the alpha map editor, which opens in the upper right section, and drag the video source icon onto the alpha map container.

Alpha map parameters:

- **Detail**, sets the detail level of the map, the lower the value the higher the level of detail.
- **Image**, drag the image you want to use creating the map, onto the drop-zone here. The image must be a valid alpha image.
- **X-size**, sets the size of the map along the X-axis.
- **Y-size**, sets the size of the map along the Y-axis.
- **Height**, sets the height of the map.
- **Texture X Offset %**, sets the offset of the texture in the X-axis.
- **Texture Y Offset %**, sets the offset of the texture in the Y-axis.
- **Texture X Width %**, stretches or compresses the texture in the X-axis.
- **Texture Y Width %**, stretches or compresses the texture in the Y-axis.
- **Patch mode:**
 - **Triangle Strip**, uses the same size of triangle polygon on the whole surface.
 - **Optimized Quads**, uses bigger triangle polygons where the surface of the displacement map is flat, thereby reducing the total number of polygons and improving performance.
- **Inverse**, mirrors all coordinates.
- **Texture Coordinates:**
 - **Repeat/Clamp**, sets if the texture is to be repeated or clamped when it is too small to fit onto the alpha map's rectangle. Clamp stretches the texture to make it fit, Repeat starts drawing the texture over again.
- **Texel Precision:**
 - **Texel:** No subpixel/subtexel correction are made.
 - **Subtexel:** There is a limited number of pixels available on the screen, if a line doesn't run through a real pixel, it must be moved to the nearest one, this introduces a positional error. If subtexel is selected the program will break up pixels into smaller sub pixels in memory so that the line can be drawn to the nearest sub pixel.
- **Smooth:** Use this parameter to smooth the look of the map, without reducing the polygon details of the map. In some cases you need the number of polygons you have to obtain the correct lightning.
- **Texture Coordinates**, decides if texture coordinates are to be created, and to which level, none, full or partially. You need this enabled to some level if a texture is applied using a vertex mapping. Texture mapping is resource consuming, so if you don't need it, keep it off.
- **Color**, sets the color of the alpha map rectangle.

Arrow



General description

The Arrow built in object allows you to create and customize an arrow. The arrow is a 2D object. By clicking on the arrow icon in a container, the arrow editor opens in the upper right window.

Arrow parameters

When you drag an arrow object onto the transformation tree, a default arrow becomes visible in the rendering window. The default arrow has two heads. At the tip of each arrowhead the reference points are placed. The left point is nr.1 and the right is no. 2. The arrow editor contains the following parameters:

- **Use LOD**, enables/disables dynamic level of detail.
- **Point 1 X**, allows you to alter point 1's x offset.
- **Point 1 Y**, allows you to alter point 1's y offset.
- **Point 2 X**, allows you to alter point 2's x offset.

- **Point 2 Y**, allows you to alter point 2's y offset.
- **Width**, sets the width of the arrow shaft.
- **Style 1**, allows you to select if the arrows end at point 1 is to have a head or not.
- **Style 2**, the same as above for point 2
- **Arrow width**, sets the width of the arrow heads.
- **Arrow length**, sets the length of the arrow head.
- **Diameter**, sets the diameter of the arrow head.

Circle



Circle parameters:

- **Use LOD**, enables/disable dynamic level of detail.
- **Corners**, allows you to decide the number of corner the circle is to be constructed by. The circle is set together by tiled triangles and the higher the number of corners is set, the more triangles are used to construct the circle. The higher the value is set the more rendering performance will be used.

Circle tips

If you disable level of detail you can set the corner value low to create some other basic geometrical objects like a triangle, pentagon, hexagon etc.

Cogwheel



Cogwheel parameters:

- **Use LOD**, enables/disables dynamic level of detail.
- **tessellation**, sets the level of detail.
- **Corners**, sets the number of corners the cogwheel is to be constructed by. The cogwheel gets one tooth for each corner, so the number of teeth changes correspondingly.
- **Height**, sets the height or width of the cogwheel.
- **Tooth height**, sets the height of the cogwheel teeth.
- **Diameter**, sets the diameter of the cogwheel.
- **Hole**, creates and sets the size of a hole in the cogwheel.
- **Bevel**, sets the degree of bevel at the cogwheel.
- **Inner Bevel**, enables or disables bevel in the cogwheel hole.
- **Show Top**, enables or disables visualization of the top.
- **Show Bottom**, enables or disables visualization of bottom.
- **Center**, allows you to select where the geometrical centre should be placed on the cogwheel, center, bottom or top.

Cone



Cone parameters

- **Use LOD**, enables/disable dynamic level of detail.
- **tessellation**, sets the degree of detail.
- **Height**, sets the height of the cone.
- **Diameter**, sets the diameter of the cone.

- **Corners**, allows you to decide the number of corners the cone is to be constructed by.
- **Show Bottom**, enables or disables visualization of the cone bottom.
- **Center**, allows you to select where the geometrical center should be placed on the cone, center, bottom or top
- **Smooth**, enables a smoothing of the cone edges.

Cube



Cube parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Size X**, sets the size of the cube in the X-axis
- **Size Y**, sets the size of the cube in the Y-axis
- **Size Z**, sets the size of the cube in the Z-axis
- **Center Y**, sets the position of the cube center along the Y-axis. You can choose between, center, bottom or top.
- **tessellation**, sets the degree of detail
- **Bevel**, sets the degree of bevel at the cube.
- **Show Top, Bottom, Front, Back, Left, Right** turns on or off the display of that element of the cube.
- **Show Size 0**, enables/disables display of the cube at height value 0. If you set height to zero, the cube is by default visible. If you then the option, the cube becomes invisible. A situation where this parameter is useful is when you create bar chart animations. In such animations you of course want the bar to be invisible when the value is zero.

Cylinder



Cylinder parameters:

- **Use LOD**, enables/disable dynamic level of detail.
- **Height**, sets the height of the cylinder.
- **Diameter**, sets the diameter of the cylinder
- **Rotation**, sets the rotation in the X-axis.
- **Angle**, sets the opening angle from 0-360°. As the value decreases below 360° an increasing angle will open in the cylinder.
- **Corners**, sets the number of corners the cylinder is to be made up from.
- **Hole**, creates a hole in the cylinder, making it into a tube.
- **Center**, sets the position for the center. You can choose between center, bottom or top.
- **Bevel**, sets the degree of bevel at the cylinder.
- **Inner Bevel**, enables/disables bevel in the cylinder hole if bevel is set at the bevel parameter.
- **Show Top**, turns off/on visualization of cylinder top.
- **Show Bottom**, turns off/on visualization of cylinder bottom.
- **Show size 0**, enables/disables display of the cylinder at height value 0. If you set height to zero, the cylinder is by default visible. If you then disable this option, the cylinder becomes invisible. If you make a pie chart visualization you will have use for this option.

Dexter



Dexter General description

Dexter is a macro based text ticker tool that generates static horizontal text crawl and vertical text rolls. The Dexter editor is organized in three sections, Runtime Control, Design options and Text.

Control



The control section allows you to start, stop, continue and reset the scroll/roll. In addition it contains buttons that allows you to select the velocity mode and speed of the ticker.

- **Start**, starts the animation
- **Stop**, stops the animation
- **Reset**, resets the animation
- **Cont**, continues the animation
- **Loop**, loops the scroller/crawl

Velocity control

The velocity control allows you to define how the speed of the crawl/roll is to be calculated.

- **Duration (all groups same)** makes all group use the same amount of time to crawl/scroll from start to end. The duration will be measured at the group containing the longest text string.
- **Duration (for each group)** makes all groups use exactly the same time. As a consequence, longer text strings will scroll faster than shorter ones.
- **Speed** makes all groups scroll at the same speed, as a consequence long message will take longer time to scroll than shorter messages. The speed unit is pixels per frame divided by 10.

Design

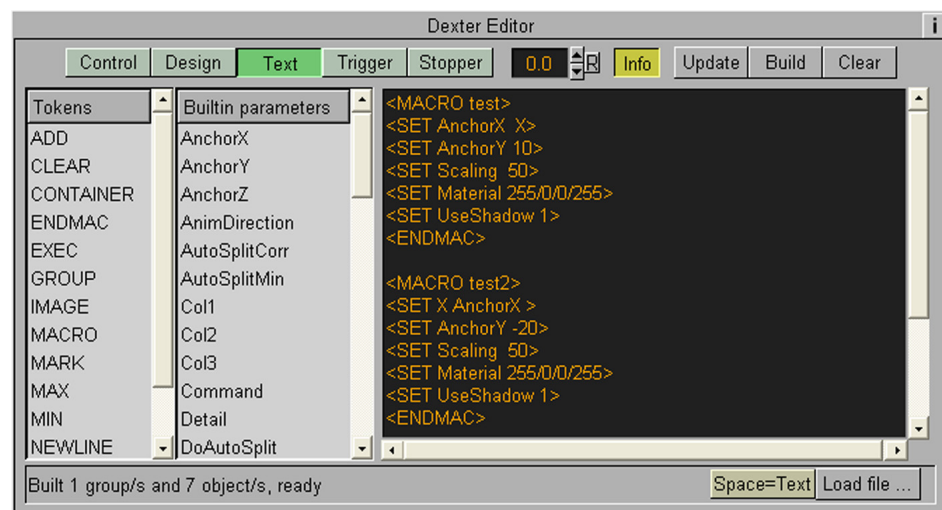


In the design section of the dexter editor, you can set a range of parameters to how the text will appear.

Dexter design parameters

- **Screen width**, sets the width of the screen used for placing the text objects and culling. Possible values range from 20 to 4000.
- **Screen height**, sets the height of the screen used for placing the text objects and culling.
- **Loop kerning**, defines the spacing between two loops. Possible values range from 0 to 1000.
- **Group kerning**, set the vertical distance between group elements.
- **Crawl mode**, creates a normal horizontal scrolling text
- **Roll mode**, creates a vertical scroll.
- **Default font**, shows the default font. To load a font, drag a font icon from the font database and onto the font dropzone. This must be set to get any text unless you specify a font in the macro and all text are executed with macros.
- **Autosplit text**: If you have a long text sting written in one line, without the autosplit feature, this is built as a big text object. This is not good for culling and performance. If you activate autosplit, long text objects will be split up into smaller ones which is better for performance.
- **Corr. val.:** Everytime the autosplit feature splits a text object, the correction value is added to the gap between the two new objects. although i read the length of a space character from the font to make a proper placing of the objects there is a slight difference in length between a split and a non-split text. therefore i inserted the correction value, to correct text that doesn't look good because of incorrect placing when the objects are split.
- **Min. object length**, sets the minimum object length for an autosplitted object.
- **col1**, allows you to set the default color #1. To set it, either drag a material icon from the material database onto the col1 icon, or use the color editor below.
- **col2**, see col1.
- **col3**, see col1.

Dexter text editor



The text editor contains the text area where you enter the script commands. In addition it contains a list of all tokens and parameters. By clicking on a token or parameter, you automatically insert it into the script text at cursor position.

- **Space=Text**, forces the parser to interpret spaces at the beginning of a line/after a token has ended as legal text (in it's default state those spaces are abandoned).

- **Load File**, allows you to browse for a script file. The default extension for the file is .edx. The script file must contain macro script with standard syntax. When the file loads, the contents is written onto the text field.

Trigger



The concept of marks and triggers allows you to send commands to Viz when the animation reaches a certain position. To make it work you need to define two elements:

- First of all you need to mark the trigger positions in the text crawl/roll. This is done using the token <MARK>.

Animation starts<MARK A>here.

You have six marks, named A, B, C, D, E and F you can put them wherever you want in the text. If you enable Show area (GUI version 1.3x under "Design" in the Dexter editor) or Info (since GUI version 1.4 next to the "Update" button) you will see a representation of the mark on screen.

In addition to the six user-definable marks there are two marks that are created automatically, named BEGIN and END. They are located at the beginning and end of the crawl/roll. Please keep in mind that the END mark is located before all the containers created for the loop feature and that marks are not looped.

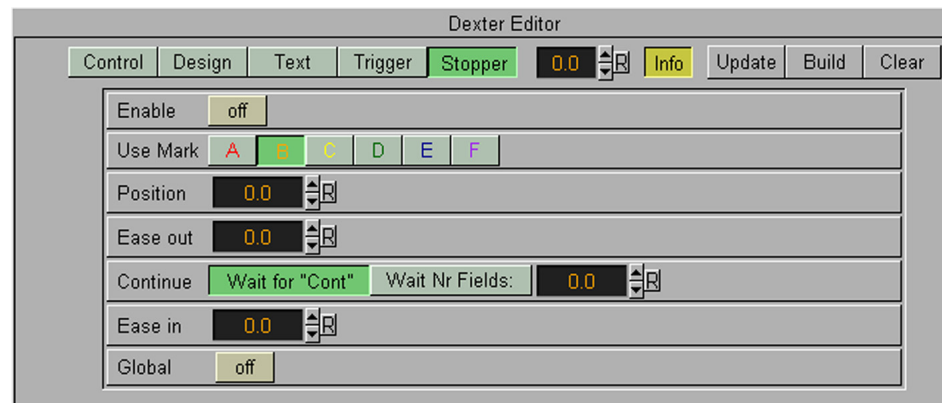
After setting marks and building the crawl/roll you must specify trigger actions. All eight marks are listed in the trigger section of the editor. The on/off switch left to the mark name is used to set that mark active/inactive. The position value right to the mark name sets the position where the triggering should occur (value 0 is at the end of the dexters bounding box. When animation is running and the mark in the text reaches this position, the specified command is triggered. If you switch on Show area/Info you will see a line representing the trigger position. Simply switch it on and watch the marks and lines interacting to see how this works. Right to the trigger position you will find a text field where you can enter the command which should be sent to Viz when a mark triggers. You may use every command available via external control, e.g. you may change values in Viz, start animations, execute plug-ins, whatever. Since version 1.32 the leading command number is optional.

In version 1.4 there are two new additions: First you may issue two or more commands when a mark reaches it's trigger position. Simply split them using a ; character. Also you may use the DEXTER location to send pseudo-commands to Dexter. Currently two such pseudo-commands are implemented:

The first one is DEXTER*STOP, which stops Dexter's animation, the second one is DEXTER*PAUSE X, where Dexter will stop the animation for X seconds (of course this is a floating point value, so DEXTER*PAUSE 0.5 is also possible) and continue with the same speed as before. Please note that these two pseudo-commands are not supported by Viz' command interface, they are only available using Dexter's trigger feature.

Since version 1.54 Dexter supports the THIS_SCENE pseudo location as well; you may want to use this to refer to objects/animations in the local scene.

Stopper



The Stopper is an extension to the triggering ability of Dexter. It allows you to create ease in/out animations at defined points along the crawl/roll. Under the "Stopper" page in the Dexter editor (since GUI version 1.6) you find all the options regarding this feature.

First you need to create a set of marks using the <MARK> token to define the stop points in your text. Then open the "Stopper" page in the Dexter editor and check the "Enable" button.

Select which marks to use. If you have defined your stop points using the <MARK C> token you need to select C here. If you enable the "Info" button you will see a representation of the animation curve in the editor. Using the position, ease in and ease out slider you can move and alter that animation like you need. (Please keep in mind that the ease out part of the animation is executed before the ease in part - although usually it's the other way round. But here we don't want to start an animation smoothly and then stop it, we want to stop it first and then restart it again).

Using the continue switch you can select what Dexter should do after the animation has stopped: Either wait a certain number of fields or wait until the Continue button in the "Control" section is pressed. If you have more than one group in crawl mode you may want to check the "Global" button as well. If it is checked all groups will be halted as soon as a marker in any group reaches the stopper position. If it is unchecked only the group that contains this marker is stopped.

Dexter Control buttons

These buttons are active in the design and text section.

- **Update**, applies the changes on the already existing containers.
- **Build**, rebuilds all objects.
- **Clear**, clears the text objects

Script programming

Tokens

Name	Syntax	Function
MACRO	<MACRO Name>	<p>Allows you to define macros. Macro definition start with the token <MACRO> and ends with <ENDMAC>. You can put tokens as well as text into your macros. They can then be executed using the <EXEC> token.</p> <pre> <MACRO Test> <SET Scaling 100> Test <SET Scaling 10> <ENDMAC> This is a<EXEC Test> </pre> <p>Macros can not be executed more than once at the same time, e.g.</p> <pre> <MACRO Nr1> <EXEC Nr2> <ENDMAC> <MACRO Nr2> <EXEC Nr1> <ENDMAC> <EXEC Nr1> </pre> <p>produces a runtime error.</p>
MARK	<MARK name>	See paragraph about triggering.
ENDMAC	<ENDMAC>	Ends the macro definition. (Nested macros are not allowed).
EXEC	<EXEC Name>	Executes the macro of that name.
SET	<SET Parameter Value>	Sets a parameter value

Name	Syntax	Function
ADD	<ADD Parameter Value>	<p>Adds a value to a parameter <ADD AnchorX 5.7> This adds the value 5.7 to the current value of the parameter AnchorX. If both the parameter and the added value are floating point numbers, an arithmetical addition is done. If not, both parameters are interpreted as text strings. The script: <SET Test Hello> <ADD Test 10.23> \$Test results in an output of the text string "Hello10.23". <SET Test 10.23> <ADD Test Hello> \$Test works in the same way: The output is "10.23Hello" in this case. But when you add two numbers, like: <SET Test 10.23> <ADD Test 54.17> \$Test the result is 64.4. Note that this result is always stored in the parameter after the addition is done. In the upper examples the value of the parameter is read out, then the addition is done and afterwards the result is written back into the parameter Test. Of course you may add two parameters as well. For that you need to use the \$ sign: <ADD AnchorX \$Step> In this case the value of the parameter Step is added to the value of the parameter AnchorX. The result is then stored in AnchorX. Subtractions are also possible: <ADD AnchorX -\$Step></p>
CLEAR	<CLEAR Parameter>	Clears a parameter; sets it to "".

Name	Syntax	Function
NOTEXT	<NOTEXT>	<p>Describes a new text object, which is separated from the previous one.</p> <p>To prevent <NEWLINE> tokens being inserted, this token could be useful. Only if Dexter's parser found text and a line feed is detected, a <NEWLINE> token is inserted. In macros without text, for example, no tokens are inserted. Possibly you could get problems with unwanted gaps in your text due to unwanted automatically inserted tokens, using <NOTEXT> you can eliminate those <NEWLINE> tokens you don't want in your text</p>
NEWLINE	<NEWLINE>	<p>The <NEWLINE> token is used to separate text objects from each other. A text object is a single Viz container which holds text, font, material ect. properties. The text objects are also used for Dexter's internal culling. This is very useful for saving performance, since long text objects eat up a lot of rendering time, especially when you're not using texture fonts.</p> <p><NEWLINE> tokens are automatically inserted when a new line in the text editor (exactly: when a line feed character (ASCII 10) is found) is started. So you don't need to type <NEWLINE> over and over again:</p> <p>Object 1<NEWLINE>Object2 means the same as: Object 1 Object 2</p> <p>The gap between two text objects is specified by the two parameters Step and ObjectKerning. While ObjectKerning specifies the general distance between text objects, images and inserted containers, the Step parameter only works for text objects.</p> <p><SET Step 200> This is a huge gap. <SET Step 20> This is a small gap.</p>

Name	Syntax	Function
IMAGE	<IMAGE Name>	<p>Inserts an image in the text. As parameter you need a valid location from the image pool.</p> <p><IMAGE test/NewsLayout/Logo2></p> <p>The image is then inserted into the text with Scaling applied (see parameter Scaling for more).</p>
MIN	<MIN Parameter value>	<p>If the parameter value is smaller than the value entered here, it sets the parameter=value. This token works like the token <MAX>, except that it stores the lower value in the parameter again.</p> <p><MIN \$AnchorX 100></p>
MAX	<MAX Parameter value>	<p>If the parameter value is bigger than the value entered here, it sets the parameter=value. Use this for a comparison between floating point values.</p> <p><MAX \$AnchorX 100></p> <p>In this example the current value of the parameter AnchorX is compared to the parameter value (in this case 100), and the higher value is then stored in the parameter AnchorX again. This token might be useful when you want to create table-like alignments of objects.</p>

Name	Syntax	Function
CONTAINER	<CONTAINER "container_location">	<p>Puts a container object in the scroller. The source container will automatically be hidden.</p> <p>The container location must be in the 1/2/3 tree hierarchy form. <CONTAINER 2/1> This means: Go to the second container on top level and take the first one under this container.</p> <p>You can drag a container from the tree and drop it onto the text pad. This will automatically insert the container in the scroll.</p> <p>When you use the <CONTAINER> token, you might probably get problems when building the text with the "Update" button. This is because containers are not completely initialized when you use that function, you need to rebuild the text using the "Build" button, then the old containers are deleted and new, empty ones are created.</p> <p>Since version 1.4 Dexter allows two additional ways to describe a container path:</p> <p><CONTAINER \$\$ANIM_LOGO> <CONTAINER #351></p> <p>With the \$ notation you can give a container name. This works exactly like in external control.</p> <p>With the # notation you can reference a container by number. Please be aware that container numbers may differ every time you load a scene. Also new since 1.4 is an addition to the syntax: Instead of writing \$\$ for inserting a \$ character, you may enclose text with " characters: <CONTAINER "\$ANIM_LOGO"></p>

Name	Syntax	Function
GROUP	<GROUP>	<p>This token allows you to split your text into groups. Groups are vertically aligned. In crawl mode you can use them to create more text crawls with independent looping or different speed, in roll mode to create lines of text and align them horizontally.</p> <p>This is the first line.<GROUP> This is the second line.</p> <p>You don't need to type <GROUP>. An empty line is converted to a <GROUP> token automatically, so the upper example can be written as:</p> <p>This is the first line. This is the second line.</p> <p>Groups are built with relative AnchorX and AnchorY coordinates and aligned after creation, according to the currently active mode (crawl or roll).</p>

Built in parameters

Name	Default value	Function
AnchorX	0	<p>When a new object (text object/image/container) is created, the x position of this object is set to the value of AnchorX. After creation of the object, the value of AnchorX is increased by the width of this object and the current value of ObjectKerning. On a <NEWLINE> token AnchorX is increased by the current value of ObjectKerning and the value of Step. The default value for AnchorX is half the value of Screen width (which is entered in the GUI). AnchorX is reset to the default value everytime a new group is created.</p> <p>Hint: Since AnchorX increases by the value of ObjectKerning everytime a new object is created, you will have a gap everytime you set a token in between your text. If you set ObjectKerning to zero, you will still find a little gap between your objects. This is because the object's alignment is based on their bounding boxes. To come around this you need to set ObjectKerning to a small negative value. The script:</p> <pre><SET Material white><SET ObjectKerning -3.3>vi<SET Material orange>z<SET Material white>rt</pre> <p>produces the word "Vizrt", where the z is drawn in a different color than the rest of the word. The value -3.3 is just a placeholder, you need to find out the correct value for the font you use.</p>
AnchorY	0	<p>This works in conjunction with the AnchorX value, it describes the y position of a newly created text object/image/container. Unlike the AnchorX value it won't be increased or decreased when a new object is created. It resets to it's default (zero) on every <NEWLINE> token. Please be aware that AnchorY is always relative to the current group position!</p>

Name	Default value	Function
AnchorZ	0	This parameter holds the Z position of every new object being created. It is set to zero before script execution starts and not reset afterwards. So if you change it, you have to set it back by yourself. Be aware that if your objects have a different Z depth, Dexter's internal culling routine probably won't handle them correctly anymore.
AnimDirection	Default	Since version 1.4 you can change Dexter's animation direction. This parameter defaults to "Default", which means: crawls are animated from right to left and rolls from bottom to top. Setting the parameter AnimDirection to any other value than "Default" (actually: any value that doesn't start with a "D" or "d" character) the direction of the animation is reversed, which means: crawls are animated from left to right and rolls from top to bottom.
AutoSplitCorr	0	Sets the autosplit correction value.
AutoSplitMin	20	Sets the minimum object length for the autosplit function.
Col1	255/255/255/255	Holds the color value Col1, form the gui/external control. To set the material to Col1 use <code><SET Material \$Col1></code>
Col2	255/255/255/255	Holds the color value Col2, form the gui/external control. To set the material to Col2 use <code><SET Material \$Col2></code>
Col3	255/255/255/255	Holds the color value Col3, form the gui/external control. To set the material to Col3 use <code><SET Material \$Col3></code>

Name	Default value	Function
Command		<p data-bbox="975 327 1402 584">The Command parameter can be used to send commands to Viz while Dexter is building objects. This should come in handy for everything Dexter can't do by itself, like extruded text for example, although the Template parameter gives you a more comfortable way of achieving this.</p> <pre data-bbox="975 591 1246 674"><SET Command - 1\$_MAIN\$_VERSION> \$Command</pre> <p data-bbox="975 680 1402 1140">The command is given in the same form as every external control command, including the leading integer number. In the example above you can see that you're not allowed to use spaces in the command string, since they are used as separation characters inside tokens by Dexter. Write \$_ instead. The answer that is returned from Viz is stored in the Command parameter, so you may read it out with \$Command. Note that since version 1.32 the leading command number is optional, so you might as well write:</p> <pre data-bbox="975 1146 1198 1229"><SET Command MAIN\$_VERSION> \$Command</pre> <p data-bbox="975 1236 1402 1431">In addition to this parameter there's the LastContainer parameter, which holds the id number of the last container Dexter has built. You can use it to add special functions, like alpha or key, to your text:</p> <pre data-bbox="975 1438 1402 1576"><SET Command - 1\$_SCENE*path_to_scene*TREE* #\$_LastContainer*EXTRU- SION\$_SET\$_EXTRUSION*Extru- sion></pre>

Name	Default value	Function
Detail	Auto	<p>Sets the detail setting of the text objects. Values are like in the text editor in the Viz Artist gui: 'Auto', '1', '2', '3', '4', '5', '6', 'T', 'B1', 'B2', 'B3' and 'B4' (case doesn't matter).</p> <p>SET Detail T> This is texture font. <SET Detail Auto> This is normal text. <SET Detail B4> This is blurred.</p> <p>Using texture fonts instead of the usual text geometries can save a lot of rendering performance.</p>
DoAutoSplit	0	<p>This parameter switches on or off automatic splitting of text objects. The longer text objects are, the less effective is Dexter's internal culling, where objects which are outside the culling area will be hidden by Dexter. To save performance it's better to split the text into smaller groups, the autosplit feature can do this for you.</p> <p><SET DoAutoSplit 1> This long, long, long text is split into pieces by the autosplit feature.</p> <p>The text is split only where a space character is found, where AutoSplitMin is the minimum number of characters a text object has to contain before a space counts as split. The text in <SET DoAutoSplit 1> <SET AutoSplitMin 7> 1234 5678 9012 is split after the character 8. In the same example, with an AutoSplitMin value of 20, the text would not be split at all. But an AutoSplitMin value of 2 would result in the text being split into 3 text objects.</p> <p>The AutoSplitCorr value is a float that is added to the x position of a split text object. This is needed since a few fonts don't cut "smoothly" and you can notice differences between a split and a non-split text.</p>
Enlighted	0	<p>This parameter controls whether a text object should be influenced by lights or not.</p>

Name	Default value	Function
FontStyle		<p>FontStyle holds the path to the current font, which is applied to newly built text objects. If you drag a font from the font pool into Dexter's text editor, this parameter is automatically inserted.</p> <p><SET FontStyle NewsFonts/Arial></p>
ImageAlign		<p>Using ImageAlign you can alter the alignment of images. Default behavior is to match the bottom lines (Y) of images and text (the bottom line of characters like a, m or x), thus the image's Y center is half the images height above the bottom text line. The X center of the image is the same position as a new text object would be placed onto. You can align the images relative to this point by specifying a direction like north, northeast, southwest, etc.:</p> <p><SET ImageAlign nw></p> <p>To set the Y center of an image to be on the bottom line use the b character:</p> <p><SET ImageAlign bse></p> <p>This would set the center Y position to match the bottom line of the text and the image to be aligned southeast to it (so the upper left corner would be at the center point).</p>
ImageOffsX		<p>ImageOffsX can be used to move the image center point (which is mentioned in the preceding paragraph) in X</p>
ImageOffsY		<p>Like ImageOffsX, for Y position.</p>
ImageOffsZ		<p>Like ImageOffsX, for Z position.</p>
Jump		<p>Using this parameter you can set a gap between two groups.</p> <p><SET RollAlign r> <SET Jump 0> Director John Dee <SET Jump 20> Producer <SET Jump 0> Jimmy Dee</p> <p>It is the equivalent of the Step parameter for horizontal alignment, thus it's called Jump.</p>

Name	Default value	Function
LastContainer		Holds the number of the last container dexter has built. You can use this parameter in combination with the Command parameter if you like to affect that container.
LastX	0	If a new text object is created, the last X position is stored here
LastY	0	If a new text object is created, the last Y position is stored here
MarkOffsX		When a mark is set via the <MARK> token in crawl mode, MarkOffsX is added to it's x position.
MarkOffsY		Same as MarkOffsY, works in roll mode and for a mark's y position.
Material	255/255/255/255	<p>Material is applied to a text object when it is created. There are two ways of defining the color value of the material:</p> <p><SET Material 255/0/127/255></p> <p>or</p> <p><SET Material design/ChannelXY/sports/footer_red_light></p> <p>The first way is to enter the color values directly in the form red/green/blue/alpha. Values go from 0 to 255. The second way is to give a path to an existing material item in Viz' material pool. This material is then copied from the pool.</p> <p>In addition you can use the color slider in the Dexter GUI for entering color values. See parameters Col1, Col2 and Col3 for that. A <SET> token is automatically entered when you drag/drop a material from the material database into Dexter's text field.</p>
ObjectKerning	5	This parameter defines the value AnchorX is increased after creating an object. Unlike Step, which is only applied when a newline occurs, this value is added everytime a new object is created

Name	Default value	Function
RollAlign	No value	<p>Using RollAlign you can align your groups vertically in roll mode. Possible values for this parameter are r, R, l and L. r and R are interpreted as right, l and L as left. Every other value is interpreted as centered. This setting can be given for every group independently.</p> <p><SET RollAlign r> This is right.</p> <p><SET RollAlign l> This is left.</p> <p><SET RollAlign djasldaldjlasjdlas> And this is centered.</p>
RollBorder		<p>When a group is aligned using the RollAlign parameter, you can specify the distance from the right or left border. When aligned in the center, this parameter does nothing.</p> <p><SET RollAlign r> <SET RollBorder 200> This is slightly right.</p> <p><SET RollBorder 100> This is more right.</p> <p><SET RollBorder 0> This is far right.</p>
Scaling	20	Sets the scaling in percent (20 means a scaling of 0,2)
ShadowDirection	320	Sets the direction of the shadow.
ShadowMaterial	204/204/204/204	Sets the material of the shadow.
SoftShadow	0	Set this to 1 to use soft shadow.
SoftShadowLevel	1	Sets the level of soft shadow from 1-4.
spacesaretext		Forces the parser to interpret spaces at the beginning of a line/ after a token has ended as legal text (in it's default state those spaces are abandoned)

Name	Default value	Function
Step	15	<p>Everytime a newline is found AnchorX is moved further by the value of Step, precisely the value of Step is added to AnchorX. This allows you to specify the spacing between text objects and works similar to the parameter ObjectKerning. The difference is that Step is only added to AnchorX when a newline occurs where ObjectKerning is added after every creation of a text object, image or container. This parameter is the horizontal equivalent of the parameter Jump.</p> <p><SET Step 0> This is near.</p> <p><SET Step 200> This is far.</p>

Name	Default value	Function
Template		<p>When you drag the Dexter plug-in in your scene there are two containers created under the Dexter container: One called "Animation" and one called "Templates". The template container can be used for setting up functions and plug-ins to be used by Dexter when creating new text objects. Let's say, for example, you want Dexter to build a certain text object with key (where the key's alpha is 35.7% and rendered in subtractive mode) and the zsort function. Now simply drag an empty group container from the function menu under Dexter's template container, drag key and zsort onto this container and set up the key function as needed. Give this container a unique name. To access the template use the Template parameter.</p> <p><SET Template Jimbo> This text object is created from the Jimbo template.</p> <p>When the next text object is created, Dexter copies the container you created from the template container into the animation and applies all other parameters like Material, UseShadow, Enlighted, FontStyle, etc. onto it. If Dexter should use no templates again you only have to clear the Template parameter: <CLEAR Template></p>
TemplateOverride	0	<p>When it is set to 1 a template overrides the font, material, etc. settings of Dexter. When set to 2 the template also overrides scaling, as well as Y and Z position.</p>
TextKerning	0	<p>Adjusts the kerning of the text.</p>
UseShadow	0	<p>When set to 1, text objects are created with their "Use Shadow" switch set to on. You can find this switch in the ont in Viz. The parameters ShadowDirection, ShadowDistance, ShadowMaterial and ShadowZOffset work together with UseShadow, as well as UseSoftShadow.</p> <p><SET UseShadow 1></p>

Name	Default value	Function
UseSoftShadow	0	When UseShadow and UseSoftShadow are set to 1, text objects are created with their "Use Soft Shadow" switch on. SoftShadowLevel defines the level of that soft shadow. Possible values are 0, 1, 2, 3 and 4.
WordSpacing	0	Adjusts the word spacing.
XScaling, YScaling, ZScaling	20	When objects are created by Dexter they are scaled, see also parameter Scaling for that. To enable independent scaling of x, y and z axis the parameter Scaling is copied to the three parameters XScaling, YScaling and ZScaling. When you write <SET Scaling 23> internally the value 23 is first written into the parameter Scaling and from there copied into the parameter XScaling, YScaling and ZScaling. This way you can access all three scaling values at once. When you write <SET XScaling 50> then only the parameter XScaling is changed, leaving Scaling, YScaling and ZScaling untouched. This way you can access all three scaling values independently. When an object is created, the parameters responsible for it's scaling are XScaling, YScaling and ZScaling. Scaling itself is actually not needed, it's simply a shortcut for changing all three values at once.

- Color values are described in the form R/G/B/A.
- Col1 is used as default if no other color is set.
- Material can also hold an existing material from the material database, e.g. /artcolors/lightgreen.

Syntax rules

The syntax consists of tokens and text. <tokens> are surrounded by <>, e.g. <ENDMAC>. spaces are used to separate tokens and arguments. Text is everything that is not bracketed. Tokens can be everywhere in the text, e.g. "Viz<SET Material 234/34/58/114>Artist" is valid. you don't need to write every token in a new line. e.g. <MACRO Green><SET Material 255/0/0/255><ENDMAC>, but nested tokens are not allowed. A macro can't execute itself.

In addition to the tokens, there is the control character \$:

Syntax	Function
\$Parameter	inserts the value of Parameter in the text or in a variable
\$_	inserts a space character in the text (you may use that in tokens)
\$<	inserts a < character
\$>	inserts a > character
\$\$	inserts a \$ character

the \$ character works as an escape character and can be placed in tokens and in the text. In text you can use it to visualize the values of parameters throughout the text string. In tokens you normally use it to parse values from a parameter into a variable.

The script:

```
text text
<SET Step 60>
text text
$Step
```

Will produce the text string: "text text text text 60"

This also works inside tokens:

```
Some Text
<SET Temp $AnchorX>
<ADD Temp -$LastX>
<ADD Temp -$Step>
<ADD Temp -$ObjectKerning>
Width of last text object was $Temp
```

\$\$

Two \$ characters are interpreted as a normal \$ character.

How to use the \$\$ character:

```
<SET Test ParameterName>
To enter a parameter value write $$$Test.
To enter a $ character write $$$$.
```

\$_

This inserts a space character into the text. You can use this in tokens if you want to assign a text with spaces to a parameter:

```
<SET Message Dexter$_is$_a$_nice$_plug-in.>
$Message
```

\$<, \$>

Inserts < or > brackets into the text, which would be otherwise interpreted as tokens.

To set mark A write \$<MARK A\$>.

Dexter basic function

A ticker sequence in Dexter can be made easily just by selecting a default font, enter a text in the text field, click the "build" text button and run the ticker by pressing the start button. You don't have to build any macros to make it work!

Execution order

The elements in the Dexter script are executed in the following order. The program will read from the top and start building the first text or the first <EXEC> element it finds. A text does not have to be lead by an <EXEC> element to be executed:

Simple example:

```
<MACRO main>
<SET AnchorY 15>
<SET Scaling 10>
<ENDMAC>
Text with no leading exec
<EXEC main>
Text formatted by the main macro
```

This script will produce the text: "Text with no leading exec Text formatted by the main macro"

The "Text formatted by the main macro" string will be placed higher than the first element since an AnchorY setting is 15.

When this text is build the program starts from the top and stops at the first executable element which is the string "Text with no leading exec", it then finds the <EXEC main> with the following text string "Text formatted by the main macro" and builds the text with the settings defined in the macro definition.

Two line example:

If you want a ticker text with two lines the script is more complex:

```
<MACRO UpperLine>
<MAX AnchorX $TempX>
<SET GroupStart $AnchorX>
<SET AnchorY 15>
<SET Scaling 10>
<ENDMAC>

<MACRO LowerLine>
<SET TempX $AnchorX>
<SET AnchorX $GroupStart>
<SET AnchorY 0>
<SET Scaling 15>
<ENDMAC>

<EXEC UpperLine>
Director
<EXEC LowerLine>
Frank Smith
<EXEC UpperLine>
Camera
<EXEC LowerLine>
Peter Smith
```

The logic of this script can be explained like this:

1. The first instance of <EXEC UpperLine> is executed.
<MACRO UpperLine>

```

<MAX AnchorX $TempX>#Anchor X is zero and TempX is zero, so the selec-
tion of the max value is the zero
<SET GroupStart $AnchorX># which now is zero
<SET AnchorY 15># which now is 15
<SET Scaling 10>
<ENDMAC>

```

The text "Director" is then started at anchorX zero. The anchorX is then displaced by the "x-space" from that text segment, as an example say the anchorX is now 12

```

2. The first instance of <EXEC LowerLine> is thereafter run:
<MACRO LowerLine>
<SET TempX $AnchorX># which now is 12
<SET AnchorX $GroupStart># which now is 0
<SET AnchorY 0>
<SET Scaling 15>
<ENDMAC>

```

The text "Frank Smith" is then started at anchorX zero. The anchorX is then displaced by the "X-space" from that text segment, as an example say the anchorX is 20

```

3. The second instance of <EXEC UpperLine> is then executed.
<MACRO UpperLine>
<MAX AnchorX $TempX> # AnchorX is now 20 and TempX is 12, so the out-
come of the function is 20
<SET GroupStart $AnchorX># which now is 20
<SET AnchorY 15>
<SET Scaling 10>
<ENDMAC>

```

The text "Camera" is then run at anchorX 20. The anchorX is then displaced by the "X-space" from that text segment, as an example say the anchorX is enlarged by 15. The new anchorX is then 35

```

4. The second instance of <EXEC LowerLine> is then run:
<MACRO LowerLine>
<SET TempX $AnchorX># which now is 35
<SET AnchorX $GroupStart># which now is 20
<SET AnchorY 0>
<SET Scaling 15>
<ENDMAC>

```

The text "Peter Smith" is then started at anchorX 20. The anchorX is then displaced by the "X-space" from that text segment, as an example say the anchorX is enlarged by 25. The new anchorX is then 45

Newline logic

The token <NEWLINE> describes a new text object, which is separated from the previous one. Every newline moves the AnchorX a bit further, depending on the value of Step. This token will be inserted internally to prevent you from typing <NEWLINE> over and over again.

The following text:

One

Two

is converted to

One
<NEWLINE>
Two

No newlines are inserted at the following situations:

- From the beginning of the script until the first occurrence of text.
- After a line that doesn't contain text.
- After the tokens MACRO, ENDMAC and NOTEXT, and the last newline before those tokens will be deleted

Example:

	<----- no newlines because no text occurred
<SET Camera Bobby>	<----- still no text
Return of the	<----- here is the first newline
Killer Tomatoes II	<----- no newline (because of token MACRO
<MACRO SetDir>	
<SET Director Jimmy>	
<ENDMAC>	
Director:	<----- here a newline again
\$Director	<----- another one
<SET FontStyle Arial>	<----- no newline, no text in this line
Camera:	<----- newline again
<NOTEXT>	<----- no newline
X	<----- no newlines until here.
	<----- newline again
<SET Material \$Col2>	<----- no text, no newline
	<----- newline again
y<NEWLINE>z	<----- the same as if written on separate lines

**Displacement
Map**



General description

The displacement map creates a topographical map based on the alpha value pattern of an alpha image.

Displacement Map parameters

- **Detail**, sets the detail level of the map, the lower the value the higher the level of detail.
- **Image**, drag the image you want to use creating the map, onto the drop-zone here. The image must be a valid luminance or alpha image.
- **X-size**, sets the size of the map along the X-axis.
- **Y-size**, sets the size of the map along the Y-axis.
- **Height**, sets the height of the map.
- **Texture X Offset %**, sets the offset of the texture in the X-axis.
- **Texture Y Offset %**, sets the offset of the texture in the Y-axis.
- **Texture X Width %**, stretches or compresses the texture in the X-axis.
- **Texture Y Width %**, stretches or compresses the texture in the Y-axis.
- **Patch mode:**
 - **Triangle Strip**, uses the same size of triangle polygon on the whole surface.
 - **Optimized Quads**, uses bigger triangle polygons where the surface of the displacement map is flat, thereby reducing the total number of polygons and improving performance.
- **Inverse**, mirrors all topographic coordinates through the zero level. This will make a mountain top to a crater.
- **Texture Coordinates:**
 - **Repeat/Clamp**, sets if the texture is to be repeated or clamped if it is too small to fit onto the displacement map's rectangle. Clamp stretches the texture to make it fit, Repeat starts drawing the texture over again when it reaches the end.
- **Texel Precision:**
 - **Texel:** No subpixel/subtexel correction is made.
 - **Subtexel:** There is a limited number of pixels available on the screen, if a line doesn't run through a real pixel, it must be moved to the nearest one, this introduces a positional error. If subtexel is selected the program will break up pixels into smaller sub pixels in memory so that the line can be drawn to the nearest sub pixel.
- **Smooth:** Use this parameter to smooth the look of the map, without reducing the polygon details of the map. In some cases you need the number of polygons you have to obtain the correct lightning.
- **Texture Coordinates**, decides if texture coordinates are to be created, and to which level, none, full or partially. You need this enabled to some level if a texture is applied using a vertex mapping. Texture mapping is resource consuming, so if you don't need it, keep it off.

Fade Rectangle



General description

The Fade Rectangles four sides can be set to fade in a way specified by the user. Both the degree of fading and the area influenced by the fading can be customized.

Fade Rectangle parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **tessellation**, sets the degree of detail.
- **Width**, sets the width of the fade rectangle.
- **Height**, sets the height of the fade rectangle.
- **Lock Blend**, allows you to decide whether to adjust the blending looked for both the X- and Y-axis or single; each axis separately.
- **Blend Range**, allows you to set the blend range if looked mode is selected.

- **Blend Range X**, if single blend is selected, you set the blend range of the X-axis here.
- **Blend Range Y**, if single blend is selected, you set the blend range of the Y-axis here.
- **Blend Factor**, sets the alpha ramp of the fade rectangle. To see the effect set the alpha value to 0 and adjust the blend factor.
- **Color**, displays the color of the rectangle. Set the color by either editor below, or drag a material icon from the material database and onto the color icon.
- **Alpha**, sets the alpha value at the rectangle edges. The blending starts from this value and blends more and more with the rectangle color along the blend range.

Globe



Globe general description

The Globe object is a version of the Sphere that has texture mapping options specially fitted for map textures. You can create a part of a sphere by using min/max latitude and longitude values. It can be used to create a round-earth representation of a geographic map.

Globe Parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Tessellation** - Sets the level of detail
- **Radius** - Sets the radius of the globe
- **West** - Crops the globe from the west
- **East** - Crops the globe from the east
- **South** - Crops the globe from the south
- **North** - Crops the globe from the north
- **Use GeoRef Box** - Crops automatically the globe to the area that is represented by the Georeferenced map. Only relevant when using a GeoRef map texture.
- **Face Camera** - Makes the cropped area face camera.
- **Texture** - sets texture mapping model
 - **Absolute** - The selected region of the globe (west to east, south to north) maps the entire texture. In this mode the texture is assumed to be unprojected and cover the degrees range (west-east, south-north)
 - **Relative** - the texture is assumed to represent the entire area of a complete globe (360x180 degrees), and is mapped accordingly on the selected region of the globe. In this mode the texture is assumed to be unprojected and cover the degrees range (-180 - 180, -90 - 90)
 - **Georef** - this mode works only with georeferenced images (those that have the little globe in their icon). A georeferenced image is mapped on the selected region of the globe using its appropriate coordinates. Any projection, supported by our georeferencing feature, will work.

Graph



Graph general description

The Graph object allows you to create a 2D or 3D graph with up to 50 values.

Graph parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Width**, sets the width of the graph.
- **Height**, sets the height of the graph.
- **Start**, sets the starting point of the graph.
- **End**, sets the ending point of the graph.
- **Line Width**, defines the line width.
- **3D**, enables/disables 3D visualization of the graph.
- **Extrusion depth**, sets the graph's extrusion depth.
- **Add backside**, enables/disables visualization of the graph backside.
- **Use vertex color**, enables a base color for the graph. Set the values in the color editor at the bottom or drag a material icon onto the small square below the parameter.
- **Line color**: If you enable use of vertex color, the a sample of the color is shown here.
- **Alpha**, if vertex color is enabled, the alpha value of the graph can be set here.
- **X0/Y0-X49-Y49**, allows you to define the different value points of the graph by giving coordinates on the X- and Y-axis.

Icosahedron



General description

An Icosahedron is a Platonic solid composed of twenty faces that span twelve vertices, each face of which is an equilateral triangle. An icosahedron can be considered a rough approximation for a sphere.

Icosahedron parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Smooth**, enables smoothing of edges.
- **Depth**, sets the number of planes at the icosahedron. The more planes you set the more it looks like a sphere.
- **tessellation**, allows you to choose level of detail setting between low, middle and high.

Pointer



General description

The pointer is similar to the arrow object. The main difference is that the pointer has a joint link, so it looks more like a "bend" arrow. The pointer has two reference points which controls the pointer's position. The no. 1 point is in the joint link and the second point is at the arrow head.

Pointer parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Point 1 X**, sets the position of point 1 on the X-axis.
- **Point 1 Y**, sets the position of point 1 on the Y-axis.
- **Point 2 X**, sets the position of point 2 on the X-axis.
- **Point 2 Y**, sets the position of point 2 on the Y-axis.
- **Width**, sets the width of the pointer shaft.
- **Cap Style Begin**, allows you to select the style on the pointer's beginning. The options are, flat, circle or arrow.

- **Cap Style End**, same as above for the end of the pointer.
- **Arrow Length**, sets the length of the arrow head.
- **Arrow Width**, sets the width of the arrow head.
- **Circle Radius**, sets the radius of the circle, if cap style is selected.
- **Circle Segments**, sets the resolution of the outer side of the pointer angle. Switch to wireframe to see the effect as you change the value.
- **Joint Style**, sets the style of the pointer joint, rounded or miter.

Rectangle



Rectangle parameters

- **tessellation**, sets the degree of detail.
- **Bevel**, enables and sets the degree of bevel at the rectangle edges.
- **Width**, sets the width of the rectangle.
- **Height**, sets the height of the rectangle.
- **Corners**, sets the number of corners the rectangle is build up from, it become visible, when you set a bevel on the rectangle.
- **Use Vertexcolors**, enables you to set the four color parameters below. If this option is set, the rectangle does not respond to container color anymore.
- **Upper left**, sets the color and alpha of the upper left section of the rectangle. Press the colored button to enable color editing on that section and change the color in the color editor below. You can also drag a material from the material database and onto the color icon.
- **Lower left**, sets the color and alpha of the upper left section of the rectangle.
- **Upper Right**, the same as above for the Upper right section of the rectangle.
- **Lower Right**, the same as above for the Lower right section of the rectangle.

RfxSmoke



General description

The RfxSmoke object is a complex object plug-in that allows you to create realistic visualization of smoke, fire, explosions and much more. (The level of creative skill is the only limitation). The basic function of the object is that it emits particles from a point and moves them in some direction until they die or fade away. By using different texture mappings and settings, many thrilling effects can be achieved.

RfxSmoke parameters

- **Number of particles**, sets the number of particles to be active at the same time. The number of particles is the product of the Birth rate and the Life Span, so the value should normally not be set higher than that. To the right of the input value field, you see "required" which is Birth Rate*Life Span. If the value is set much lower than the required, the particles will create puffballs instead of a continuous smoke.
- **Birth rate**, sets the birth rate of particles per second.
- **Life span**, sets the life span of the objects in seconds. The field to the right of Life span, allows you to set a percentage, over which the life span of the particles will vary. If you have set Life Span to 3 and the variation to 50%, the life span of particles will vary randomly between 1,5 and 4,5

seconds. This will help create more realism, since particles in real smoke or fire do not have an equal life span.

- **Reference**, allows you to choose the particles reference to either "Container" or "World". If "Container" is chosen, after being emitted the particles will relate their position to the container's position. If you move the emitter object, the particles will alter their position correspondingly. If "World" is chosen, the particles will maintain their position and path even if the emitter is moved, they have their own "World coordinates".
- **Billboard**, if you enable billboarding for the particles, the particles will maintain a frontal position against the camera, when it moves.
- **Spawn**, opens a section of parameters relating to the position of the smoke emitter. You can define the shape and size of the area in which the emitter should emit the particles.
 - **Type**, The buttons marked "Cube" and "Sphere" allows you to set whether the spawn area should be shaped as Cube or a Sphere.
 - **Size**. The X-, Y- and Z-values sets the size of the selected form. If you have selected "Sphere" and you set only the X-value, the emitter will create particles while it moves randomly along a line. If you set an Y-value in addition, the emitter will move within a circle, and finally if you set a Z-value, within a sphere. If you have selected a "Cube" the same applies but the "stages" would then be a line, a rectangle and a Cube.
 - **Spawn Range**, sets the range within the Sphere or cube, the emitter should use for distributing the particles. Default is 100% where it uses the whole spawn area. If you reduce the value, a section starting from the middle of the spawn area, where no particles are being emitted, will increase correspondingly. If the value is set to 1%, the emitter only emits particles at the outer edge or surface of the Cube or Sphere.
- **Impulse**, opens a sub section where you can set parameters on how the emitter emits the particles.
 - **Impulse Speed**, sets the speed the particles have when they are being emitted. The field to the right of "Speed", allows you to set a percentage over which the particle speed will vary
 - **Rot X**, sets the X-axis angle of the "opening" in the emitter; the emitter hole.
 - **Rot Z**, sets the Z-axis angle of the "opening" in the emitter; the emitter hole.
 - **Angle X**, sets the X-axis opening angle of the emitter hole. It can be done locked with the Z axis or independently. The button to the right of the value field enables/disables locking of the axes.
 - **Angle Z**, sets the Z-axis opening angle of the emitter hole.
 - **Twist**, creates a twist of the opening angles of the emitter hole. It can also be described as a rotation of the emitter around the Y-axis.
 - **Align to center**, disables all the above impulse parameters and aligns the particles above the emitter.
- **Forces**, opens a sub section with parameters for creating environmental effects, like wind, gravity and Air Resistance.
 - **Gravity**, sets the degree of gravity affecting the path of the particles. If it is set high, the smoke will go downwards. With a negative gravity the particles will rise faster.
 - **Air resistance**, sets the degree of air resistance. It creates a force that, to a level you define, prevents the particles from rising.
 - **Turbulence X, Y,Z**, allows you to define a simulation of turbulence on the particles as they rise. The turbulence effect is achieved through a jittering of the particles. The X-, Y- and Z-values sets the axis of the jittering movement.
 - **Wind Force**, sets a level cross wind. A positive value makes it blowing from the left to towards the right, a negative the other way around. Use the two next parameters to adjust the angle of the wind if you don't want a horizontal wind force.

- **Wind Rot X**, sets the rotation of the wind force on the X-axis. Altering this value and the value below, will change the direction of the wind defined in the "Wind Force" parameter.
- **Wind Rot Z**, sets the rotation of the wind force on the Z-axis
- **Size**, opens a section with parameters to the size of the particles at the moment of their birth and death. Between the time for birth and death, the size is a product of a linear interpolation.
 - **Size Birth X,Y**, sets the size of each particle at the time it is being emitted.
 - **Size Death X, Y**, sets the size of each of the particle at the time it fades away and dies.
- **Rotation**, opens a section where you can define a static rotated position or a spin of the particles. Both the static rotated position and the spin can either be set as an offset equal on all particles or as a range where the particles will be rotated or spun randomly. To easily see the effect, set the "Number of particles" to a very low value and alter the rotation parameters.
 - **Initial offset**, sets the degree of rotation the particles should have. This rotated position is static and will remain throughout the life of each particle.
 - **Initial range**, sets a degree range of rotation the particles should have. This rotated position is static and will remain throughout the life of each particle. If f.ex. the value 30 is set, the static rotated position of the particles will vary randomly between 30° and -30°.
 - **Spin offset**, sets the degree of spin each particle should have throughout its life time. The higher the value, the higher the number of spins. A positive value creates an anti clockwise rotation, a negative creates a clockwise.
 - **Spin range**, sets a spin degree range for the particles. They will spin randomly between the parameter value and the same value "mirrored" through zero.
 - **Follow path**. If the RFXSmoke object is animated, enabling this option will make the particles align their position to the animation path in the same way as is the case with the "Follow path" option in the Transformation Editor.
- **Color**, allows you to set the color of the particles throughout their life cycle. The particles can be set to change their color through stages from their birth to their death. The color editor section contains a rectangle that displays the color range through the particles life cycle. The left side is at birth time and the right is at death time. By default there are two "color keyframes" at each end, that you can click on to set color. The color range between two keyframes will be an interpolated transformation. More keyframes can be added to create a more complex color transformation between particles birth and death.
 - **To set a color**, select a keyframe by holding the mouse above it. It will change its color from black to white to indicate that it's selected. When you single click on it, it will change color to red to indicate that it's in editing mode. You can then set the desired color in the color editor below.
 - **To add a new color keyframe**, click at the wanted place in the rectangle displaying the color range. A new color keyframe is then created and you can set the color. You can move it with the mouse afterwards.
 - **To delete a color keyframe**, select it and press either the delete- or backspace button.
- **Alpha**, allows you to set the alpha values of the particles throughout their life cycle. The alpha editor contains a curve with points that you can move in X and Y to change the form of the curve. The X-axis is the life time of the particles and the Y-axis is the alpha value.
 - **To move a point**, select it by holding the mouse above it. It will then change its color from black to white to indicate that it has been selected. When you single click on it and holds the button down, it will

change color to red to indicated that it's in editing mode. You can then move it in the wanted position to alter the alpha curve.

- **To add a new point**, click on a "free" section of the curve
- **To delete a point**, select it and press the delete button
- **Reset Particles**. Clicking this button removes all the particles already emitted.
- **Show Forces**. Enabling this option displays "help lines" that shows the forces settings of some of the parameters.
- **Freeze Motion**, halts the emitting process.

ImageFX



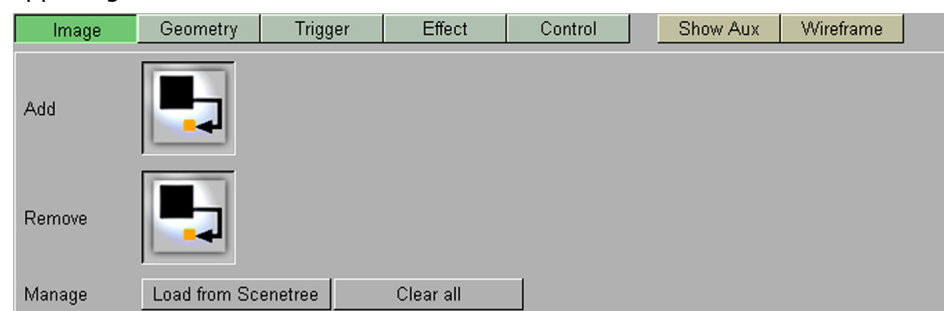
General description

Viz|imageFX is a tool that enables you to create a wide variety of transitions between one or more images. The plug-in uses many different effect models that can be customized by the designer through parameters. Some properties use advanced mathematical formulas to create the effects, and not all of them can be explained meaningfully in normal words. Because of this, not all properties can be explained completely here, one have to "play" with some properties to see the effect they create in a given situation (what some properties are set to will have impact on the effect that other properties give).

How does it work

The images are split into a number of user-defined sections and every section is controlled independently based on progress maps. Any number of images of any Viz|Artist supported format can be controlled using ImageFx. All images will be resized (if needed) to the first image size. The image transition order can be modified on the fly and images may be added to the sequence using external commands.

The plug-in is a object plug-in and you find it in the object database in the "built in" section along with the standard primitives. Drag the ImageFX plug-in onto the scene tree and click on the imageFX icon. The editor opens in the upper right window.



The editor has a row of buttons at the top that opens the main properties sections.

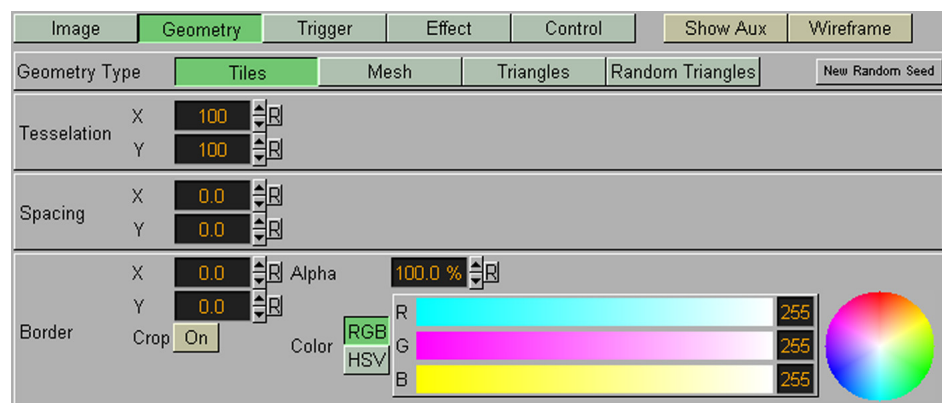
Image

Add the images you want to create transitions between by dragging them from the image database and dropping them onto the "Add dropzone"



The added images will be visible in the scene tree as subcontainers of the ImageFX plug-in container. It is possible to add images directly as subcontainers of the plug-in container, but to make the plug-in recognize the added images, you must press the "Load from scenetree" button. Remove images by dragging them from the image database and onto the "Remove dropzone" or remove them directly from the scene tree.

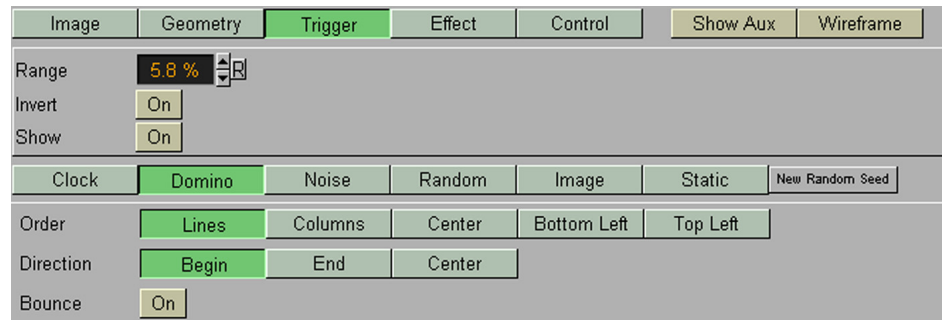
Geometry



The image is split into n stand-alone sections that can be modified independently.

- **Geometry type** - Select here which kind of geometrical primitives you want the images to be split up in.
 - **Tiles** - GL quads, the number of quads are X tessellation times Y tessellation.
 - **Mesh** - (vertices) GL quad strip, the number of vertices are (X tessellation +1) times (Y tessellation +1). Modification can apply to every vertex in the mesh.
 - **Triangles** - GL triangles, the number of triangles are X tessellation times Y tessellation times 2. Modification can apply to one triangle (3 vertices).
 - **Random Triangles** - GL triangles, Triangles are generated using random points. Number of triangles is a minimum of X tessellation times the Y tessellation. Modification can apply to one triangle (3 vertices).
 - **New Random Seed** - is relevant if you have chosen Random triangles. It specifies a seed for the random number generator. Even though the program use random numbers, the layout of a specific random seed will always look the same. Press the wireframe button and click the new random seed button and you will see the effect.
- **Tesselation** - sets the degree of detail. (Enable wireframe to view the effect).
- **Spacing** - Sets a spacing between the geometrical parts in the image.
- **Border** - If spacing has some value, this parameter will draw a border between the elements.
- **Alpha** - Sets an alpha value for the border.
- **Color** - Sets a color for the border.

Trigger



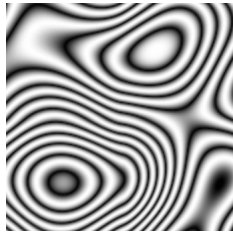
In the trigger section you can select between different progress maps that defines how every section of the image is to be triggered by the global progress. Each section of the image receives its base value, the global progress is a value that changes as you animate by using one of the options in the control menu. Since every section gets a different base value it will be triggered differently by the global progress.

Example: In a domino effect which starts at the bottom left the section at the bottom left will receive a value of 100 and the section at the top right will receive a value of 1. As the progress moves section with high values will be influenced faster than sections with low values. if we continue with the above example with a fade effect the bottom left section will fade out when progress is 20 but the top right section will only fade out when progress is 100.

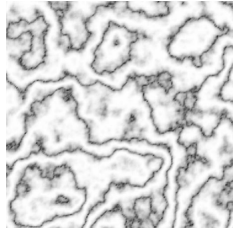
The options for progress maps are:

- **Clock** - The values for the sections are computed based on a clock hand movement. It has the following properties:
 - **Center** - Sets the center for the clock hand movement
 - **Start angle** - Set the starting angle for the clock hand
 - **Whirl** - Applies a whirl effect to the movement
 - **Direction** - Sets the direction for the clock hand movement.
- **Domino** - The values for the sections are computed based on a domino movement. It has the following options:
 - **Order** - Sets the order in which the tiles will be moved. Choose between Lines (Line by line movement), Columns (Column by column movement, Center (From center and outwards movement), Bottom Left (Bottom left tiles first and then diagonal movement towards top right corner) or Top Left (Top left tiles first and then diagonal movement towards bottom right)
 - **Direction** - Sets the starting point of the movement.
- **Noise** - The values for the sections are computed based on user-defined noise maps created using noise algorithms (Perlin noise). The settings create a noise texture and the image transformations are based on this image. When you edit the noise properties it is a good idea to enable the "show" button. This will let you see the image that are created and you will see changes you do instantly
 - **Wave, Cloud, Marble and Noise** are predefined noise parameter settings. It is normally a good idea to start with one of these and thereafter edit until you get the desired effect.

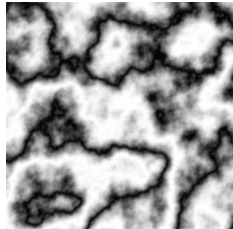
The four images below show how the four noise parameters look like when the "show option" is enabled:



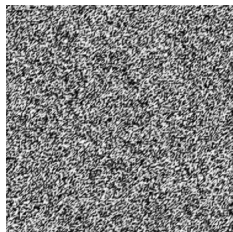
Wave



Marble



Cloud



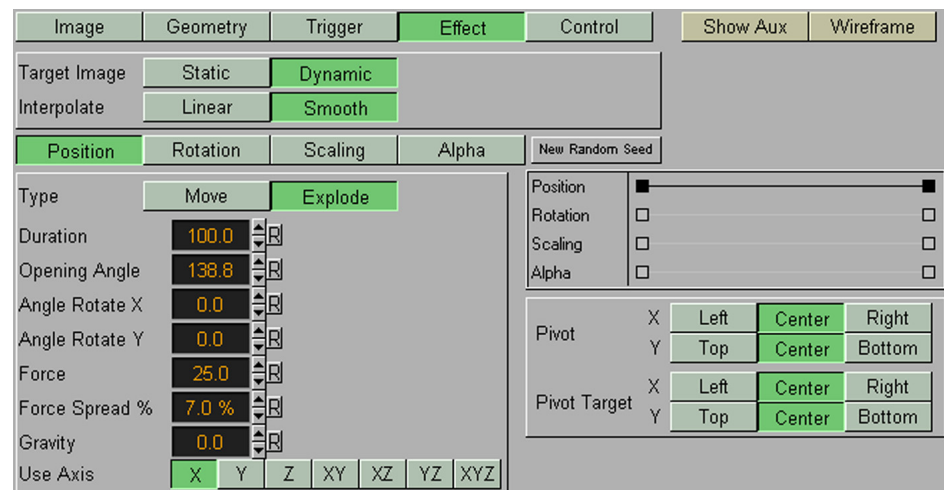
Noise

- **Turbulence 1-3** - are noise functions. These must be "played with" to see what kind of effect you get. They influence each other. So what effect one gives depends on the state of the other.
- **Frequency** - Define something that can be described as the "wave length" of the noise function. A large number mean a small "wave length" which will give a lot of changes in the image, while a small number will give a smoother image more simple image.
- **Exponent** - Controls the ratio between the white areas and the black areas without changing the shape significantly. Exponent zero give a full white picture, and as the value will grow, the black area will become larger and larger.
- **Scale X/Y** - Are quite similar to the usual texture scale, but they are working with reverse logic. In the usual texture scaling, increasing the scale is like zooming into the image, but in image-fx it works like zooming out. Also the units are different, but beside this, they are quite similar.
- **Random** - The values for the sections are computed randomly.
- **Image** - The values for the sections are computed based on pixel values of an image. What the function does is to create links between the animated image tiles and the pixels of the image and then use the luminance

level of each pixel to set the base value for the tiles. By default 100% luminance sets high base value and 0% sets a low base value. This option allows you to generate any progress map you want by using other tools such as Photoshop

- **Static** - All sections receive the same value. The complete image will move out and the other will move in.
- **New random seed** - Is relevant if you have chosen random order. It specifies a seed for the random number generator. Even though the program use random numbers, the animation for a specific random seed will always look the same.

Effect



The Trigger section defined the order in which the tiles would be influenced by the global process. The effect section allows you define what kind of effect/transition that will be used.

- **Target image**
 - **Static** - The target image is not effected at all by the effect. A typical example would be a reveal effect where target image is "hidden" behind main image and a transition effect on the main image will reveal the target image.
 - **Dynamic** - The target image is controlled by the effect. A typical example would be main image flying out of the frame and at the same time target image flying in. If a dynamic target image is used, its effect can be either being the invert of main image or can be defined separately.
- **Interpolate**
 - **Linear** - A linear interpolation will be used for the transformations.
 - **Smooth** - A smooth interpolation will be used for the transformations.
- **Position>Move** - You can here specify a position for the source image, for the target image and a random position.
 - For the source image, which is the image that is currently displayed, you specify a source position. This is the position that the image tiles will move to when the animation runs. If no target position is specified, the target image will animate from the source position to the initial position of the source image. If you specify a target value the target image will move from that position and onto the initial position of the source image.
 - **Absolute** - All sections will move to the same end position.
 - **Relative** - The sections will maintain their original relationship.
- **Position>Explode** - Final positions for the sections are calculated base on a user-controlled explode algorithm. The tiles of the image are treated as particles in an explosion and they are moved in one to three axis depending on what you specify. All tiles are thrown out of an emitter that

can be defined to have any opening angle, so the particles can be sent within a narrow angle, like a canon fire, or in all direction like a explosion. The parameters are:

- **Duration** - Sets the duration for the movement of the tiles/particles.
- **Opening angle** - Sets the opening angle for the emitter. Remember to choose more than one axis to get any clear effect.
- **Angle Rotate X** - Rotates the emitter hole on the X-axis.
- **Angle Rotate Y** - Rotates the emitter hole on the Y-axis.
- **Force** - Sets the degree of force that throws the particles out in space.
- **Force spread%** - randomizes the initial impels of the tiles. Spread 0 % means all elements have the same impulse, 100% means a high degree of randomness.
- **Gravity** - Sets a gravity force for the environment.
- **Use Axis** - Select which axes or axis to use.
- **Rotation** - The sections are rotated from degrees 'A' to degrees 'B'.
- **Scaling** - The section is scaled from scale 'A' to scale 'B'.
- **Alpha** - The alpha value of the sections are changed from value 'A' to value 'B'.

To all options (except alpha) a certain degree of user-controlled randomness can be added.

- **New random seed** - Is relevant if you have a random property. It specifies a seed for the random number generator. Even though the program uses random numbers, the animation for a specific random seed will always look the same.

Mini-stage



The final effect of transformation can be a combination of effects where the timing between the different effects can be tuned using the mini-stage. Move the start and end keyframes to time the effects.

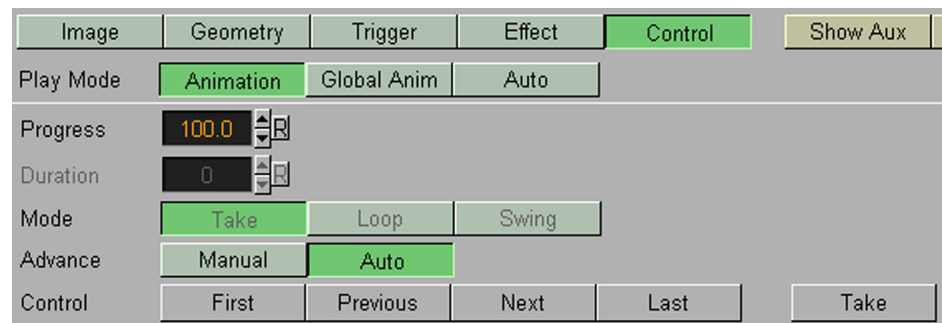
Example: A position explode effect can be combined with a rotation effect so the sections will rotate as they explode. An alpha effect can be added to fade out the sections at the end.

Pivot

Pivot	X	Left	Center	Right
	Y	Top	Center	Bottom
Pivot Target	X	Left	Center	Right
	Y	Top	Center	Bottom

The pivot properties have relevance when you use rotation on the tiles. The settings decide where the rotation center should be on the tiles. To see the effect, create a low tessellation transition with tiles and a 180 degree X or Y rotation. Change the pivot properties for Y pivot if you have an X rotation or the X pivot if you have an Y rotation and see the effect.

Control



The global progress value that is used to animate the sections can be controlled in different ways:

- **Auto** - When using auto mode ImageFx will start the effect as soon as the "Take" button is invoked. Useful for external control in order to run a sequence of images.
- **Animation** - The global progress value is animated and the effect follows the stage progress. In this mode you can combine the effect animation with other animations.
- **Global animation** - When using a sequence of images the complete progress will animate all images.

Noggi



The Noggi geometry plug-in is a kind of parallelogram with bevel settings for each corner. It can be drawn as an outline and it can be extruded.

Noggi parameters

- Width - sets the width of the object.
- Height - sets the height of the object
- Bevel (all four corners) - Sets the bevel for the corners
- Stretch - moves the top and bottom in opposite directions thereby stretching the figure.
- Horizontal alignment
- Vertical alignment
- Geometry type - Choose between an outline or an solid object
- Inner Outline - When the object is set to "outline" and the object is extruded, you must set this parameter to either Clockwise or Counter-clockwise to get the right face orientation.
- Outline Width - Sets the width of the outline.
- Edge Points - Sets the number of edge points. A high number will give smoother bevels.
- Tessellation - Not implemented yet
- Crop - If enabled, you can crop the four faces of the object with the parameters below
- Crop Face (All four faces)

Nquad**General description**

The Nquad plugin starts out as a rectangle but can be formed to a parallelogram, a trapezoid or a rhombus. It can also be set to draw only a border geometry.

Nquad parameters

- **Use LOD** - enables/disables dynamic level of detail.
- **Tessellation U and V** - sets the degree of detail in the two directions.
- **Orientation** - switches the orientation from Top-Bottom to Left-Right.
- **Top/Left Width** - sets the width on the Top/Left part of the figure.
- **Bottom/Right Width** - sets the width on the Bottom/Right part of the figure.
- **Height/Width** - sets the height or width depending on the Orientation.
- **Depth** - sets the depth.
- **Center X** - switches the center in X between Center, Top and Bottom.
- **Center Y** - switches the center in Y between Center, Left and Right.
- **Shearing** - sets the shearing value, so the bottom is shifted.
- **Image Mapping** - may be set to Stretch or Tile.
- **Crop Top/Left** - crops the top.
- **Crop Bottom/Right** - crops the bottom.
- **Border Geometry** - if switched on only an outline of the figure will be drawn.
- **Top Border Width** - sets the width on the top part of the border.
- **Bottom Border Width** - sets the width on the bottom part of the border.

SplineFX plug-ins

The SplineFX plug-ins are set of Viz|Artist geometry plug-ins, that enhances the modeling capabilities for 2D curved objects. While producing the plug-ins, a special emphasize has been given to offer the user an easy and interactive direct manipulation of the geometry in the Viz|Artist rendering window. The SplineFX set includes 2 plug-ins, namely 2DRibbon and 2Dpatch. The geometry of both plug-ins is 2D. The transformation matrix of the container can be changed, also on any parent container, without creating problems for the interactivity.

2DRibbon

The 2DRibbon is a curved strip. It is ideal for making customized curves to symbolize frontiers on a map or similar. The way the object is curved is easily edited in the object's editor. To create a 2DRibbon object, open the object database and from the "built-in" section drag the 2DRibbon object onto the scene tree. It is possible to enable interactivity so you can drag the counter-points of the ribbon by using the mouse. The object's editor has the following parameters:

- **Number of control points** - Sets the number of control points along the ribbon.
- **Texture Direction** - Define which direction of the texture that is to be mapped along the ribbon.

- **Ribbon Length, Ribbon Width, and Reset** - These properties are useful for giving initial values for the control points. The values of Ribbon Length and Ribbon Width will take affect after the Reset button has been pressed.
- **Constant Width** - With this option enabled, the width of the ribbon is fixed. The value that defines the width is then the width of the first control point.
- **Reset button** - After initially having set the length and width of the ribbon, press the reset button to apply the changes. Note! The control points will be reset back to a aligned setup, so the length and width settings should be set as the first step in the workflow.
- **Texture Head Location** - Defines the position on the ribbon where the edge of the texture is mapped. The unit are percentage of the ribbon length.
- **Texture Length** - Defines the portion of the ribbon, in percentage, that the texture will be mapped onto.
- **Resolution** - Defines the number of subdivisions to polygons between the control points.
- **Control Point Size** - The size of the control points, in the Viz|Artist normal units.
- **Interactive Mode** - A toggle to enable or disable the interactive mode of direction manipulation with the mouse.
- **Editing Range %** - Sets the amount of "effect" the movement of one point (move with shift button pressed) should have on the neighboring points
- **Editing Mode** - When you move a point, you can by holding the shift button pressed as you move, move the other points simultaneously. How they move depends of which editing mode you choose.
 - **Constant** - The other points will move their positions exactly equal to the one you move.
 - **Linear** - The other points move in a linear way, based on the distance from the point that is being moved.
 - **Exponential** - The other points move in an exponential way, based on the distance from the point that is being moved.
- **Draw control points** - Toggles on/off the display of the control points
- **X, Y, and W values** - Sets the X/Y position and the width of the curve at the different control points. These values will typically will be changed through editing directly in the Viz|Artist rendering window with the mouse.

Workflow

Below follows a description of a typical workflow for creating and defining a 2Dribbon object.

1. Drag the 2Dribbon object from the Viz|Artist built in objects
2. Set the right number of control points, according to your estimation of how complex ribbon you are going to create. This number can be changed at any time. The plug-in will redistribute the new points along the already defined ribbon.
3. Estimate the final length and width of the ribbon, enter it to the right fields in the GUI, and hit the Reset button. You will get a straight ribbon with the right sizes.
4. Set correctly the toggle of the Constant width according to your requirements.
5. Move the ribbon to the right place in your scene. If it is going to be on some horizontal plan, it might be easier first to edit it in the default orientation, and afterwards rotate it to the right place. There is no problem to continue working on it after the rotation has been made, but because of the perspective, it might be less straightforward.
6. Turn on the Draw Control Points toggle and the Interactive toggle. While working with the mouse, you don't have to be very close to the control

- point that you would like to manipulate. The plug-in will find by itself the nearest control point to where is your mouse is. The selected control point will get yellow color, as compared to all the rest that are white, to indicate that it has been selected for editing.
7. Note that you will be able to do the direct manipulation just if the object is the selected object in the Viz|Artist container tree. Remember also, that regardless of the position of your mouse, if the ribbon is the selected object, and the interactive toggle is on, always one control point will become selected and will be changed while working with the mouse. Hitting the space key on the keyboard deselects the object currently selected, and then the mouse gets back its usual functionality until the ribbon object is selected again.
 8. With the left mouse button, according to the selection roles above, you can change the X and Y location of each of the control points. With the right button you can change the width of the ribbon. According to the Constant Width toggle the change will be just on the control point, or all over the ribbon. All three values are interpolated with cubic spline, and accordingly the affect of moving a control point is global, but with decreasing affect with the distance from it along the ribbon.
 9. Once the shape of the ribbon is more or less defined, it is a advisable to set the resolution parameter. Performance wise you should set it as low as possible, but it should be high enough to guarantee a smooth enough look in the corners of the spline, and also a smooth enough mapping of the image/texture on the ribbon.
 10. Choose an image and drag it on the ribbon object in the usual way. After doing it open the image GUI and set the Mapping property to Vertex, and the Wrap property to Clamp.
 11. The next step is to map more accurately the image on the ribbon. Set first the Texture Length according to your estimation, then choose the Texture direction, and then play with the Texture Head Location to see the texture flowing along the ribbon. This is a good point to review again all the parameters you set before.
 12. The last step is to define an animation if necessary. Typically with the ribbon the animation will be just on the Texture Head Location.

2DPatch



The 2DPatch is a two dimensional planar curved grid of polygons. The grid is defined by control points that are located on the perimeter of the grid. The parameters of each one of the control points is the location X, Y, and the assigned texture coordinates U, V. The calculation of the internal grid points is done in with a mix of a one-dimensional cubic spline and a two-dimensional bilinear interpolation. It is possible to specify the interpolation direction to be just in one direction, or in both. Depending on the shape that should be created, it is important to choose the right direction. (For example it is not too difficult to create with this plug-in an annulus, after choosing the right interpolation direction along the radius.)

- **X and Y Number of Intervals** - Number of intervals between the control points in the X and Y direction respectively.
- **Interpolation Direction** - Defines the direction of interpolation from the control points on the perimeter to the internal grid points.
- **X Size, Y Size, Reset XY, and Reset UV** - This part is useful to give initial values for the control points. The values of X-Size and Y-Size will take affect just after hitting the Reset XY button. The Reset UV button is setting default values to the texture coordinates.
- **X Resolution and Y Resolution** - Defines the number of subdivisions to polygons between the control points in the X- and Y-direction respectively.

- **Interactive Mode** - A toggle to enable or disable the interactive mode of direction manipulation with the mouse.
- **Editing Range %** - Sets the amount of "effect" the movement of one point (move with shift button pressed) should have on the neighboring points
- **Editing Mode** - When you move a point, you can by holding the shift button pressed as you move, move the other points simultaneously. How they move depends of which editing mode you choose.
 - **Constant** - The other points will move their positions exactly equal to the one you move.
 - **Linear** - The other points move in a linear way, based on the distance from the point that is being moved.
 - **Exponential** - The other points move in an exponential way, based on the distance from the point that is being moved.
- **Draw Control Points** - A toggle to draw or not to draw the control points.
- **Control Point Size** - The size of the control points, in the Viz|Artist normal units.
- **X, Y, U, and V values** - Sets position and texture coordinates for each of the control points. These values will typically will be changed just in the Viz|Artist rendering window with the mouse.

Workflow

The workflow is quite similar as for the 2DRibbon plug-in. See that section for details. There is one special detail for the 2DPatch plug-in: To change the control points for the texture mapping, use the right mouse button.

General

Warning!

While defining an animation on the parameters of a SplineFX object, you must make sure that the right GUI window is open in the Viz|Artist GUI. Without it the changes that you are doing with the mouse will not take effect with regard to the animation. Be especially careful on this point because of the first Known Bug mentioned below.

Tips

1. It is sometimes useful to work with the SplineFX plug-ins while in wire-frame mode. If you have a texture with alpha, in order to see it correctly you need to turn the image off with the small enable/disable button near the image icon.
2. Remember that if the object is not selected, the interactive mode is not active. As result of this, the first click with the mouse on an unselected object will be always with the normal Viz|Artist functionality of the mouse, of moving, rotating, etc. the object.
3. If you are done with the interactive session of defining a SplineFX object, turn the Interactive toggle to be off.
4. While working on a SplineFX object in the interactive mode, it is not possible to select another object with the mouse in the Viz|Artist rendering window. The way to select another object is first to hit the space bar on the keyboard, and thereafter select the other object.
5. Doing animation on the SplineFX object is quite expensive performance wise. Be aware to it, and try to use as much as possible a small number of control points and polygons.
6. For the texture used for the 2DRibbon plug-in, make sure that along the flow direction, you have on both texture edges a clean line with alpha equal to 0. Other wise you will get a wake of the image edges in front or behind it.
7. Do not change the number of control point along an animation. It is possible, if necessary, to change the resolution number.

8. While defining an animation, make sure to remove from it, if necessary, the toggle values of Show Control Points, Interactive Mode, etc.

Known limitations:

1. Note that when you have in the scene one of the SplineFX objects, by choosing an object in the Viz|Artist rendering window, the Viz|Artist GUI is not updated automatically. This is in fact true for all objects and not just the SplineFX objects.
2. While coping a SplineFX object, in order to make it work in the interactive mode, it is necessary to save the scene first.
3. For the 2Dpatch object: If the texture coordinates on the control point are getting outside the [0,1] range artifacts may appear. Be aware to keep it within the [0,1] range.

Ring



Ring parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Corners**, sets the number of corners on the ring.
- **Inner radius**, sets the size of the inner radius.
- **Outer radius**, sets the size of the outer radius.
- **Rotation**, rotates the ring like a “turning wheel”. This is typically used in combination with an open angle, to place the angle at the desired point.
- **Angle**, defines an open angle on the ring.

Roll



General description

The roll object allows you to create a rectangle that rolls up in different ways. This is typically used for different kinds of unveiling. To make the roll look realistic, the Expert function must be added to the container to enable visualization of backface and twosided lightening.

Roll parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Size X**, sets the size of the rectangle on the X-axis.
- **Size Y**, sets the size of the rectangle on the Y-axis.
- **tessellation**, sets the level of detail.
- **Angle**, sets the angle of the rolling up.
- **Diameter**, sets the diameter of the roll.
- **Helix factor**, defines how tight the roll is to be rolled up. With a high helix factor, there will be much space between the layers of the roll.
- **Drawing**, selects how the roll rectangle is to be drawn up. If you switch to wireframe mode, and toggle between the two settings you will see the difference:
 - **Segments**, draws the rectangle up by using many long lines stretching from edge to edge. This mode does not look as good as Mesh but it demands less performance of the render-engine.
 - **Mesh**, draws the rectangle using many small equally sized triangles. The number and size depends on the level of tessellation selected. This drawing mode can be heavy to render if the tessellation is set high. Mesh creates a better lightning of the roll rectangle.

- **Style**, sets the style of the roll:
 - Cigar roll, makes the rectangle roll up in a “normal” manner.
 - Wipe, rolls the carpet up without creating a roll.
- **Outline**, enables an outline view for the roll. You must in addition add the expert function and enable the outline setting there.
- **Time**, sets the time for the roll sequence. Animate this value to create an animation of the roll object.

Soft Clip Draw Pixels



The Soft Clip plugins

Soft clip is an viz plugin that can show AVI clips, either projected on a texture, or rendered directly.

The softclip plugin comes in two variants:

1. SoftclipDrawPixels, which is a geometry plugin and draws pixels directly into the frame buffer.
2. SoftclipTexture, which projects the video on a texture.

Soft Clip Draw Pixels general description

Choose a video file to play. Playback can be achieved by either animating the frame numbers or by turning on the Live Playback. If you want the video to be a part of the 3D universe in your scene or to map the video onto an object, you must use the texture variant of the Soft Clip plugin: “[Soft Clip \(Texture\)](#)” on page 153.

Codec support

The plugin decodes Audio Video Interleave (AVI) clips using the Video For Windows Application Programming Interface (VfW API). The AVI format in itself is just a file structure relies on Coder-Decoder (CODEC) components to perform actual video decoding. This means that the SoftClip plugin is limited to supporting the AVI codecs for VfW that are installed on the machine.

Some supported codecs:

- Uncompressed AVI
- HuffYUV
- Xvid
- Pinnacle DV25

CPU and I/O bandwidth

The performance of the SoftClip plugin is most often limited by CPU and I/O bandwidth resources. Some codecs are CPU intensive, while other codecs are I/O bandwidth intensive.

E.g. uncompressed video requires little CPU work, while it puts pretty hefty demands on the I/O subsystem when playing large clips. Various forms of MPEG 4 codecs are examples that compress well and usually doesn't stress the I/O bandwidth. However, these codecs can be CPU intensive.

The SoftClip codec works best on multi-CPU machines, which allows viz to live on one CPU and the video decoding thread to live on a separate CPU.

Loading multiple scenes

One thing you should be aware of is that Viz keeps all the scenes and plugins you load in memory. So if you load 30 different softclip scenes, viz will have 30 scenes and 30 softclip plug-ins with 30 threads active. If you also specify that each plugin should have 10 preloaded frames, then you will have 300 uncompressed decoded frames preloaded into memory. This can take a substantial amount of memory and the combination of having that much data loaded and that many threads running, is likely to affect performance. An invisible softclip plugin still consumes resources, and can affect the rest of the system in the following manner:

- If a valid filename has been specified, then a decoding thread will be active.
- If the pre-load queue isn't full, it will make load images until the queue gets full.
- The plugin will respond to external input, such as frame number changes, play on/off even if it is not visible.
- The pre-load queue of the plugin will use the memory needed to keep the fully decoded frames of the queue.

To get the softclip instance to release most of its resources, set the clip name to an empty string. This will remove any thread, decoding queue, and decompressor that has been allocated for a clip.

Half-height and Draw-Pixels

When running viz in half-height mode, you need to be aware of the following: Since `SoftClipDrawPixels` by design draws pixels directly in the frame buffer, it will draw the pixels in normal aspect even if the rest of the scene was drawn half-height. In the interlaced output, the video lines from the AVI file will directly correspond to the lines of each individual field.

This will result in the video being shown in double height when you lace the field pairs together from the video output.

To avoid this, you have three options:

1. Turn off half-height rendering.
2. Use the texture variant of `SoftClip` that does not bypass the regular rendering mode of the scene.
3. Prepare the video clip for use in half-height rendering. This means that you should separate out each frame of the video into two separate fields, thereby halving the y-resolution and doubling the frame rate.

There is a potential problem with option 3. There is no way today to guarantee that the first frame of a video clip that is played back will start at a specific field, the plugin therefore can't ensure deterministic field dominance behavior.

Ring buffers

When using the softclip with a ring buffer, you should turn frame skipping off.

Soft Clip parameters

- **use LOD** - Enables dynamic level of detail.
- **Clip File** - The video file to play
- **Live Play** - in live playback mode, the frames will be displayed synchronized to viz rendering, or according to the Frames Per Second (FPS) setting, depending on the play mode that is chosen.
- **Frame number** - This is visible if Live Play is disabled. Animate this value to create a video playback.
- **Play mode**

- **Synchronized** - Will synchronize the frame rate of the AVI to the output framerate of Viz|Engine, i.e. for each rendered viz frame, the video will advance exactly one frame.
- **Frames per second** - In FPS playing mode the plugin will control frame rate by correlating the value to the rendering frame rate. This will only work if rendering rate is synchronized with video output.
- **Frames per second** - Sets the frame rate.
- **Loop video** - Loops the video
- **Use decode thread** - Choose whether to use threading for video-decoding. On multi-processor machines this can gain you performance, but threading makes scheduling less predictable and may cause viz frame rate to be less stable on a single CPU machine.
- **Preload frame count** - Sets the number of frames that should be pre-loaded when the
- **Frame skipping** - When on, viz will never wait for decoded frames, i.e. video frames will be skipped rather than forcing viz to drop or delay frame rendering.
- **Force Opaque** - When showing videos where the codec leaves the alpha channel blank, the video rendering may become completely transparent. This option will fill the alpha channel, thus making the video completely opaque.
- **Preferred CPU** - If you have more than one CPU, you can choose which one the decoding process should use: 0=any
- **Priority** - Sets priority for the decoding process.
- **X Position** - Sets the X position
- **Y Position** - Sets the Y position
- **Z Position** - Since this Soft Clip variant draws its pixels directly into the frame buffer, it has no real 3D Z-position. Changing this value just sets the Z-sorting for the soft clip container in relation to the other objects in the scene.

Sphere



Sphere parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **tessellation**, set the degree of detail for the sphere.
- **Rotation**, allows you to rotate the sphere around its Y-axis.
- **Open Angle**, allows you to create an open angle in the sphere.
- **Texture**, allows you to select between three different positions of the texture mapping coordinates when a vertex texture is mapped onto the sphere. To visualize the effect of the different settings, add a plain texture onto the sphere, like a chessboard image, and set the texture mapping to vertex in the texture editor. Create an opening in the sphere and change the sphere texture parameters. Increase and decrease the opening angle with the different settings enabled, and the effect should be visible.
 - **Absolute**, compresses and expands the texture mapping coordinates as the opening of the sphere changes.
 - **Relative**, cuts/adds texture coordinates as the opening changes. The cut/add is done on the one side of the opening angle. The coordinates are anchored at the other side.
 - **Centered**, does the same as the relative mapping, but cuts/adds coordinates from both sides. The texture is anchored in a centered position on the sphere. The center of the sphere is 180° from the point where the opening angle starts.

Spring**General description**

The spring object creates a spiral spring. Be aware, this object can easily be very heavy to render if a high level of tessellation and corners are selected.

Spring parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **tessellation**, sets the degree of detail for the spring
- **Corners**, sets the number of corners the spring should have.
- **Gradient**, sets the size of the spring gradient.
- **Diameter torus**, sets the diameter of the springs.
- **Diameter cross section**, sets the diameter of the spring cord.
- **Show Bottom**, shows/hides bottom of the spring.
- **Show Top**, shows/hides Top.
- **Center**, sets the location of the spring center, either Center, Bottom or Top.

Torus**Torus parameters**

- **Use LOD**, enables/disables dynamic level of detail.
- **tessellation**, sets the degree of detail for the torus.
- **Corners**, sets the number of corners on the torus.
- **Radius**, sets the radius of the torus from its center in the middle of the hole.
- **Radius cross section**, sets the radius of the torus cross section or "tube".

Wall**General description**

The wall object is a curved wall. The size and curving of the wall can be customized.

Wall parameters

- **Use LOD**, enables/disables dynamic level of detail.
- **Height**, sets the height of the wall
- **Width**, sets the width of the wall
- **Depth**, sets the depth of the wall curving. The higher this value is set, the more curved the wall will appear.
- **Angle**, sets the angle of the wall's curving. Use this parameter together with "Depth" to achieve the wanted curving of the wall.
- **Corners**, sets the number of internal corners the wall should be built up from.
- **Center**, sets the center axis of the object. Choose between, center bottom or top.

Wave**General description**

The wave object allows you to create a wave sequence on a flat surface. The properties of the sequence are visualized in a graphical display in the wave editor.

Wave parameters

- **Size, X and Y**, sets the size of the rectangle where the waves appear.
- **Geom Type**, allows you to select between circular and linear waves.
- **Stretch**, allows you to customize the form of the circular waves. By altering the stretch X and Y values, you can for instance obtain an oval or a compressed form of the waves.
- **Time**, is the timeline for a wave sequence. You see a thin red vertical line moving in the graphical display as you alter the time value. You can also alter the value by dragging the red line with the mouse.
- **Tess.**, sets the tessellation/level of detail.
- **Angle**, sets the angle of the waves when a linear mode is chosen.
- **Ref.Pnt, X and Y**, sets the position of the wave starting point.
- **W. Length**, sets the length of the waves.
- **Damp mode**, enables damping of the waves. The amplitude of the wave is reduced by the "damp length distance" calculated from the center of the wave. The distance is set under the parameter "Damp length".
- **Damp Length**, sets the length the distance that is used to damp the waves in the "Damp mode" function.
- **Offset**, sets an offset value of the ref. points.
- **Ampl**, amplifies the waves, resulting in both bigger and higher waves.
- **Inverse off/on**, inverts the wave curves.
- **Envelope generator**, enable this option if you want to draw the wave manually. When disabled the wave is built by a sinus wave function.
- **Position X and Y**, allows you to edit the position values of the wave control points. Select a point by clicking once on it. Its color changes to red to show that it is selected. Move it by holding down the left mouse button and move the mouse.
- **Lock**, allows you to decide whether the points controlling the crest of the waves should be edited in a locked manner with the points controlling the troughs of the waves or conversely if they are to be edited separately.

Edit by using the mouse

The control points of the spline curve, that shows the wave sequence, can be edited by using the mouse. Click on a point and drag it to the desired position.

Container plug-in functions**Grid****General description**

The Grid function is an effective tool for arranging a set of containers into a grid or tabular structure. To do this, create a group container and add subcontainers where all objects have the same position. By adding the Grid function to a group container, a tabular distribution of the objects can be defined by

setting the number of rows and columns they are to be distributed over and thereafter set the spacing between rows and columns. A common way to use this plug-in is to create a group container, create another within, and set the shape, material and animation you want on that container. The container can have its own subcontainers. By creating replicas of the "source container" and then position them by using the grid function on the group container, a homogenous tabular setup with equal properties for all containers within the group, have been achieved very easily.

Parameters

- **Number of rows**, sets the number of rows the containers are to be distributed over.
- **Number of columns**, sets the number of columns the containers are to be distributed over.

The product of the two above parameters should normally be the same as the actual number of containers. If it's higher, all rows will not be equal, if it is lower, some containers will not be shown, but they will still be rendered.

- **Row offset**, sets the distance between the rows.
- **Column offset**, sets the distance between the columns.

Circle Arrange



Circle Arrange general description

Circle Arrange is a tool for arranging a set of containers into a circular structure. To do this, create a group container and add subcontainers where all the objects have the same position. A typical way to do this is to create and position a container, set the shape, give it the material and animation you want, and thereafter make copies of it. By adding the Circle Arrange function to the group container, the function will position all the subcontainers to make a circular structure. The structure can be modified with a set of parameters. NB! The order of the containers in the group decides their placing in the circular structure. If more than one row is selected, the first containers end up in the outer circle and the last containers in the inner circle.

Circle Arrange parameters and functions

- **Number of elements**, allows you to define how many elements the structure is to be constructed by.
- **Number of rows**, sets the number of rows/rings for the circular structure
- **Distribution**, allows you to define how the containers are to be distributed in the circular shape.
 - **Monospace**, sets the same number of elements in each circle row. The spacing between objects in the outer rows will then be bigger than in the inner rows.
 - **Equal**, distributes the objects between the rows so the distance between the objects in the different rows are equal. The number of containers will then be larger in the outer rows.
- **Inner Radius**, sets the radius from the center to the first container ring.
- **Outer Radius**, sets the radius from the center to the outer ring.
- **Start Angle**. Setting this above -180° will create an open section in the circle structure. The opening is created in a clockwise direction.
- **End Angle**. Setting this below 180 will create an open section in the circle structure. The opening is created in an anti clockwise direction.
- **Orientate**. If this option is enabled, the containers will orientate itself with their bottom towards the center of the circular structure.
- **Rebuild Containerlist**. Use this button to rebuild the structure after having added or removed containers from the group.

Time Displace



Time displace general description

The Time Displace function allows you to use a grayscale/alpha image for setting animation time offsets for all the objects in a group container. This is typically useful if one wants to create some kind of randomized animation effect. What the function does is to create links between the animated containers and the pixels of the grayscale image and then use the alpha level of each pixel to set the offset of the containers. The range from 100% alpha to 0% alpha on the pixels sets the level of offset. By default 100% alpha sets no offset and 0% sets maximum offset default.

Time displace parameters

- **Image:** Drag the alpha image you want to use for creating the containers offsets, onto the dropzone here.
- **Invert**, inverts the link between the image pixels and offsets. 0% gives no offset and 100% gives maximum.
- **Image Width**, sets the height of the alpha image.
- **Image Height**, sets the width of the alpha image. Image width*Image height should normally be the same as the number of containers that are to be used in the function.
- **Pixel reference**, defines the way the pixels should be associated to the animated containers. You can choose between:
 - **Order in Tree:** The containers order in the transformation tree decides how the pixels should be associated.
 - **Name of container:** The container name sets the order in which the containers will be associated with the pixels.
- **Search Tree**, sets the search level of the function when it decides which subcontainers to include in the function:
 - **One Level**, only the first level of subcontainers is being included in the function.
 - **Recursive**, all levels of subcontainers are being included.
- **Threshold Action**, is executed if the pixel value is lower than the threshold value.
 - **None**, no action is performed.
 - **Skip Pixel**, the pixel is skipped in the offset process.
 - **Visible**
 - **Invisible**
- **Threshold**, sets the Threshold value where one of the above selected actions will be performed.
- **Stage Range (frames)**, sets the range of the offset distribution. A high value will result in bigger differences in offset between containers that are associated with different pixel alpha values.
- **Stage Offset (frames)**, adds a static offset value to all containers.
- **Filter Type**
 - **None**
 - **Including**
 - **Excluding**
- **Filter String**

Coco



Coco general description

The Coco function allows you to create copies of a container with great ease. Just add the function onto the container you want to copy, enter the wanted number of copies and click the execute button. You can alter the naming and numbering and tree level placement of the copies. This function is typically used together with arrange functions.

Coco Parameters

- **Number of Copies**, sets the number of copies to be created when the function is executed.
- **Scene Tree level**, allows you to select the hierarchic position of the copies.
 - **Same**, sets the copies on the same level as the source container.
 - **Down**, sets the copies as subcontainers to the source container
- **Clear Transformation**, sets all transformation values on the new copies, like scaling, position and rotation, back to zero. All copies will then be stacked at a initial position with all transformation values set to zero.
- **Starting index**, allows you to alter the starting point of the index numbering of the copies. By default the numbering starts on 1.
- **Container name**, allows you to change the name the containers will be given by default.
- **Execute**, after having set the required parameters, click this button to execute the duplicating process.

Colin



Colin general description

The Colin function allows you to align a container in the scene using two other as a reference. This makes it easy to maintain a good and controlled symmetry in the scene. To use the function, add it onto the container you want to align in the scene. The Colin editor has two container dropzones. Drag and drop the two containers you want to use as reference for the aligning, onto a dropzone each. The container is then aligned between the two referential containers.

Colin parameters

- **Container 1**, is the dropzone for the first referential container.
- **Container 2**, is the dropzone for the second referential container.
- **HOffset %**, sets the horizontal offset between the referential containers in percent. Horizontal means here an invisible line between the referential containers regardless of the actual boning of the line.
- **VOffset**, sets the vertical offset from the horizontal line.
- **ZOffset**, sets the Z offset from the line between the referential containers.
- **Align container rotation**
 - **2D**: With this option enabled the container will position and rotate itself in the X- and Y-axis to keep itself aligned on the invisible line between the referential containers as they move.
 - **3D**: With this option enabled the container will position and rotate itself in the X-, Y- and Z-axis to keep itself aligned on the invisible line between the referential containers as they move.

Cora**Cora general description**

The Cora function allows you to sort a set of subcontainers after many different criteria. It is a useful tool when your scene starts getting complex and it is difficult to maintain the overview. To make sorts by different criteria can help you reorganize the scene tree structure. Add the function to the container that holds the subcontainers you want to sort, select the sort criteria and press the execute button. This function used together with the Time Displace function enables many creative possibilities

Cora parameters

- **Criteria:** Selects the wanted sort criteria.
 - **Name**, performs a case sensitive sort. In normal sort order, capitalized letters are listed first. Be aware, use letter case consistently if you want to have a correct result when you perform a sort by name.
 - **X**, sorts by X-values
 - **Y**, sorts by Y-values
 - **Z**, sorts by Z-values
 - **Vis**, sorts all the visible containers first.
 - **Alpha**, sorts by the alpha value among those containers who have an alpha function attached. Alpha values set on the material of a container are ignored.
 - **Key**, sorts containers by key function. Among those with key signal attached, the key alpha value is used for the further sorting.
- **Mode**, sets the order of the sorting. Choose between setting the highest or the lowest values first.
- **Random Seed**, is relevant if you have chosen random as your sort criteria. It specifies a seed for the random number generator. Even though the program use random numbers, the animation for a specific random seed will always look the same.
- **Random Weight**, defines how random the randomize function will sort. Random 1.0 means random, random 0.0 does nothing and random 0.5 gives you a 50% chance if a container is resorted or stays in place.

Corena**Corena general description**

The Corena function renames all subcontainers of the container you apply it to. Add the function to the container which holds the containers you want to rename. Enter the new name, set the indexing parameters and press the execute button.

Corena parameters

- **Name**, sets the name you want to rename the subcontainers to. You can define the number format of the new names by entering a #-symbol in addition to the name for each digit you want in the numbering. If you enter ###newname, the containers will be named: 001newname, 002newname etc.
- **Add index**, enables/disables creation of a numbering index.
- **Starting index**, sets the number you want the numbering index to start from.
- **Reverse index**, if enabled, creates the index in a reversed order.

VCF_Param VCF_Param general description

The VCF_Param plug-in together with the VCF scene plug-in allows you to create a seamless interpolated transition from a virtual camera flight /VCF) to a real camera- and vice versa. This is only a relevant function to set up if you have purchased the virtual set expansion components.

Prerequisites

- You must have a real camera with data tracking enabled which is set in remote mode in the camera editor. In case you have several tracked cameras, the virtual camera will interpolate its position to the real camera that is selected on-air.

How to

Start by dragging the VCF scene plug-in onto the setup\plug-ins dropzone. It contains two parameters:

- **Tracking object (POS):** Defines the position or eyepoint of the virtual camera flight.
- **Tracking object (ROT):** Defines the direction of the virtual camera flight (where the camera is looking).

Keep the default values T_POS and T_ROT or change them to what you prefer.

Create a new group in your scene and add 2 new objects under it (i.e. cubes). These are to be the objects that will define the virtual camera flight. One will define the position, the other the direction. Name the containers according to the names you entered in the VCP scene plug-in (i.e. T_POS & T_ROT). Press initialize to finish.

Apply the VCF_PARAM container plug-in on the container that holds the position object.

Animate your virtual camera flight using the 2 objects to define position, and direction. Do NOT switch the whole container invisible - because the animation will not run. You may switch the objects to be invisible, at any time. It is not possible to animate the roll of the camera

In the VCF_PARAM plug-in there are two parameters that both can be animated:

- **Real Camera %:** Defines the percentage the real camera influences the position of the virtual camera.
- **Zoom:** Sets the actual zoom (FOV-Y) value of the virtual camera

You can define by animating the Real camera % parameter in what frame you want to have the real camera at 0% visibility and when at 100% visibility. (the longer the animation is the smoother the interpolation will be- 100 F should be good) For example: if you start with a virtual camera the first keyframe (0) should have 0% of the real camera, and the last should be 100% of the real camera. The virtual camera will generate a full key (masking the FG) If you choose Setup->Background->Virtual Set.

Make sure you FG object/talent are not seen while switching to the real camera. They should appear in the frame after the switch.

2D Follow



2D Follow general description

The 2D Follow function works together with the 2D Ribbon object. See, ["2Dribbon" on page 119](#). What the function basically does is to create automatically an animation path that follows the form of a 2D Ribbon object. A 2D ribbon object must be created first.

2D Follow setup

To set up the 2D Follow plug-in, do the following:

1. Create the 2D Ribbon object.
2. Add the object/objects you want to animate along the 2D Ribbon's path as a child container to the 2D Ribbon container.
3. Add the 2D Follow plug-in to this container. The object will now move when you change the Path position value. Animate this value to get the animation into the stage.

2D Follow parameters

- **Path Position** - Sets the position for the object along the path of the 2D Ribbon object. Animate this value to get an animation in the stage.
- **Translation Offset** - Sets an offset between the object and the 2D Ribbon path.
- **Keep Orientation** - If enabled the orientation between the object and the 2D Ribbon path is kept as the object moves along the path.
- **Orientation Offset** - Offsets the orientation between the object and the 2D Ribbon path.

Bar and Bar Values



Bar and Bar Values general description

These plugins allow you to build bar charts without many complex design steps. The Bar Values plugin is placed on a group container. Any subcontainer of this that holds a bar plugin will be regarded as part of the chart and can be controlled by the bar values editor. The bar containers must hold some kind of geometry that can be controlled by the bar values plug-in.

Bar plugin

The Bar plug-in tells the Bar Values plug-in that the container is part of the bar chart and it also tells which elements of the geometry on the container to control. Each container that is to be a bar, must hold a geometry and a Bar plug-in.

Bar parameters

- **Standalone** - If the Bar plug-in is going to be used without a Pie Values plug-in to set the values, enable this button. It will enable you to set the value directly on the plugin and you can define normalization factors between the input values and the value that is being set on the geometry.
- **Bar Min/Max** - If in standalone, these values set the minimum and maximum values for the bar. These values are connected to the Obj Min/Max values and by adjusting these together you can get the desired effect between input values and visual effect.
- **Obj Min/Max** - Maps the input values to the native values on the object being controlled by the plug-in. Adjust these together with Bar Max/Min on the Bar Values plugin or on the Bar plug-in itself if in standalone mode, to get the desired effect between input values and visual effect.
- **Param Type** - Sets the type of value to control on the object.

- **Param** - Controls a named parameter on the object. Use the console to find the correct name for the parameter: Enable "Show commands", Open the object's editor, change the parameter and see in the console what the parameter name is.
- **Position** - Sets a position value on the object's container. Select with axis/axes to position.
- **Rotation** - Sets a rotation value on the object's container. Select which axis to rotate
- **Scale** - Scales the object.
- **Alpha** - Controls the alpha value on the container.
- **Text** - Controls the text on the container.
- **Initialize** - Click this button to trigger changed parameters.

Bar Values plug-in

The Bar Values plugin communicates with all subcontainers that hold a Bar plug-in and lets you set values for each of them.

Bar Values parameters

- **Num Bars** - sets the number of bars to control. Should correspond with the number of subcontainers holding the bar plug-in.
- **Bar Min/Max**- These values set the minimum and maximum values for the bars. These values are connected to the Obj Min/Max values on the bar plugins and by adjusting these together you can get the desired effect between input values and visual effect.
- **Scale** - Scales all bars.
- **Bar Animation** - Reveals and animate the bars one by one.
- **Bar Value 1-25** - Sets the values for the bars.

Pie Values and Pie Slice



Pie Slice and Pie Values general description

The Pie Slice and Pie Values plug-ins lets you build a pie chart in a few design steps:

1. Add a group container and name it, for instance "pie".
2. Add PieValues to the group.
3. Add a Cylinder object as a subcontainer to the group and add a material and a PieSlice plugin. The Cylinder becomes invisible. Name the container "slice" or similar.
4. Open the PieValues editor and increase value 0 a bit. You will see that the cylinder becomes visible and its size changes.
5. Open the Cylinder editor and set Centre to "Top"
6. Rotate the pie group approximately 90° on the X-axis to make the cylinder top face the camera.
7. Add a font as subcontainer to the slice container. Give it a material that makes it visible against the background.
8. Open the text editor and set the horizontal alignment to Centre. Open the transformation editor and rotate the font around the X-axis to -90° and set Y position to 1.0.
9. Scale the font down a bit so it fits over the slice.
10. On the slice container, open the PieSlice editor and set Control Text Values to a data type that fits your needs. If you choose Integer or Float you will see that the slice text changes and displays the value for the slice.
11. In the Pie Slice editor, adjust the Text Offset to position the value label relative to the centre of the pie.

12. Now you have a single pie slice ready. Make the number of slices you want by making copies of the slice container and change their material so they can be differenced from each other.
13. Set the values in the PieValues editor to test the pie chart.

Pie Slice

The Pie Slice plugin is placed on each container that is a slice of the whole pie. It is controlled by the Pie Values plugin that must be placed on the group node above.

- **Obj Min/Max** - Maps the input values to the values on the cylinder object. Adjust these together with Bar Max/Min on the Pie Values plug-in to get the desired effect between input values and visual effect.
- **Offset** - Offsets the slice from the center of the pie chart.
- **Text Offset** - Offsets the text label from the center of the chart.
- **Control Text Values**
 - **No** - No text control
 - **Integer** - Show slice value as integer
 - **Float** - Show slice value as float (two decimals)
 - **Formatted** - uses the formatNumber plug-in to create advanced formatting of the text.
- **Initialize** - Click this button to update after parameter changes has been made.

Pie Values

The Pie Values plugin controls up to 20 pie slices. All containers with a Pie Slice plugin that is put below the container holding the Pie Values plugin will receive values from the numbered value fields.

- **Pie Min/Max** - These values set the minimum and maximum values for the pie slices. These values are connected to the Obj Min/Max values on the pie slice plugins and by adjusting these together you can get the desired effect between input values and visual effect.
- **Min Value** - Sets the minimum value of the whole pie chart
- **Max Value** - Sets the maximum value for the whole pie chart
- **Scale** - Scales all values proportionally.
- **Slice** - Reveals the slices one by one.
- **Value 0-19** - Sets the value for the pie slices.

TfxAlpha



TfxAlpha general description

The TfxAlpha function is a text effect that creates a fade in effect for the text characters. The effect sequence can be set to go many different ways. A fade in and out effect can also be achieved. To use the function add it onto a container which holds a text.

Parameters

- **Progress %**, 0% progress is the beginning of the effect, 100% the end. Animate this value from 0% to 100% to see the effect or from 100% to 0% to animate the effect backwards.
- **Progress Type**

- **Absolute:** 100% progress will animate all characters of a text, regardless how many characters it has.
- **Relative:** 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction,** sets the direction of the text effect sequence, you can choose between the following options:
 - **Left:** Starts with the first character.
 - **Right:** Starts with the last character
 - **Random:** Uses a random order.
 - **Static:** All characters are processed at the same time.
 - **Wave:** Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center:** Starts the effect from the center of the text.
 - **2 Center:** Starts the effect at the same time from the beginning and the end of the text. They meet at the centre.
- **Interpolate,** choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range,** defines how many characters that are to be in the effect processing at the same time. If f.ex. the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %.** This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence or in other words how wide the wave top is to be. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed,** specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Multibyte fonts, you should uncheck this option.
- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Alpha Begin,** sets the alpha level of the character at 0% effect.
- **Alpha End,** sets the alpha level of the character at 100% effect.

TfxArrange



TfxArrange general description

The TfxArrange function allows you to arrange characters in either a circular or a wave shape. The characters can be animated on the selected shape by animating the "Offset" value. To use the function add it to a container which holds a font.

Parameters

- **Offset**, moves the characters on the shape. 100% means one full rotation of the text on the circle.
- **Scale**, Scales the text on the shape. The parameter does not scale the characters itself, but the kerning of the characters.
- **Shape**, changes the shape of the character layout. The options are a circle or a wave.
- **Rotation**, rotates the characters around the characters X-Axis.
- **Align**, rotates the characters on the Z-axis to align the X-axis with the tangent of the shape at the characters position.
- **Circle Parameters:**
 - **Diameter**, sets the diameter of the circular shape
 - **Positioning**, defines the position of the text on the circular shape. "Relative" means that the text spacing of the text object is maintained. "Absolute" means that the text is evenly distributed out on the circle.
 - **Direction**, sets the direction of the text on the circle to either "Clockwise" or "Counter clockwise".
- **Wave Parameters:**
 - **Length**, sets the amplitude/wave length. A high value creates many and small waves, a low value fewer and bigger waves.
 - **Width** The width of the wave.
 - **Height**, sets the height of the waves without altering the number of waves as the "Wave length" parameter does.
 - **Shift**: The wave usually starts at the height 0. The shift value moves that point, thereby moving the texts position on the wave structure. Setting the Shift value to 100% brings you back to the beginning.
 - **Damping**, can be used to modify the amplitude of the wave curves.

TfxExplode



TfxExplode general description

The TfxExplode function creates an explosion like function where the characters get thrown away from their initial position. The speed, direction and spread of the moving characters can be altered with parameters. To use the function add it onto a container which holds a font. To create an animation, animate the progress% value. Other values can be animated as well.

Parameters

- **Progress%**, 0% progress is the beginning of the effect, 100% the end. Animate this value from 0% to 100% to see the effect or from 100% to 0% to run the effect backwards.
- **Progress Type**
 - **Absolute**: 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative**: 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction**, sets the direction of the text effect sequence, you can choose between the following options:
 - **Left**: Starts with the first character.
 - **Right**: Starts with the last character
 - **Random**: Uses a random order.
 - **Static**: All characters are processed at the same time.

- **Wave:** Starts with the first character, animates the effect from 0-100% and the down again to 0%.
- **Center:** Starts the effect from the center of the text.
- **2 Center:** Starts the effect at the same time from the beginning and the end of the text. They meet at the center.
- **Interpolate,** choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range,** defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %.** This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed,** specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space -** Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.
- **Ignore Newline -** Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left -** Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Duration,** defines the duration of the progressing for each of the characters. If you want a longer progressing, increase this value. You could achieve the same by making the gravity stronger and animating the progress slower, but it is more easy to increase the duration instead if your explode effect is too short.
- **Opening Angle,** sets the angle for the spread of the characters. 0 will send them straight up, 360 will spread them in a circular shape.
- **Angle Rotate X,** rotates the opening angle around the X-axis.
- **Angle Rotate Y,** rotates the opening angle around the Y-axis.
- **Force,** sets the force that throws away the characters. A high force will make them go far away, conversely a low force will create only a small motion of the characters.
- **Force Spread %,** sets a variation of the force among the characters.
- **Use Axis,** allows you to select on which axis or combination of axes the characters are to spread along.
- **Gravity,** sets a gravity force that influence the path of the characters to end up going downwards. The higher the value is set, the faster each character will divert from its initial path and start going downwards.
- **Use Rotation,** with this option enabled, the characters will rotate as they are being thrown away from their initial position. The degree of rotation is set by the next parameters, "Rotation Force"
- **Rotation Force,** sets the degree of rotation as the characters are being thrown away.
- **Show Force,** with this option enables, lines showing the characters path and speed will be visible in the rendering window.

TfxJitterAlpha



TfxJitterAlpha general description

The TfxJitterAlpha function creates an jittering motion of the characters by randomly changing the alpha value of each character. The degree of change and the start sequence of the jittering can be altered. To use the function add it onto a container which holds a font. To create an animation, animate the progress% value. Other values can of course be animated as well.

Parameters

- **Progress %**, 0% progress is the beginning of the effect, 100% is a constant jittering of all the characters.
- **Progress Type**
 - **Absolute**: 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative**: 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction**, sets the direction of the text effect sequence, you can choose between the following options:
 - **Left**: Starts with the first character.
 - **Right**: Starts with the last character
 - **Random**: Uses a random order.
 - **Static**: All characters are processed at the same time.
 - **Wave**: Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center**: Starts the effect from the center of the text.
 - **2 Center**: Starts the effect at the same time from the beginning and the end of the text. They meet at the center.
- **Interpolate**, choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range**, defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %**. This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed**, specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.
- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Alpha Begin**, sets the alpha level of the character at 0% effect.
- **Alpha End**, sets the alpha level of the character at 100% effect.
- **Randomness**, sets the intensity of the jittering alpha changes.

TfxJitterPos**TfxJitterPos general description**

The TfxJitterPos function creates an jittering motion of the characters by randomly changing the position of each character. The degree of position change and the starting sequence of the jittering can be altered. To use the function add it onto a container which holds a font. To create an animation, animate the progress% value. Other values can of course be animated as well.

Parameters

- **Progress %**, 0% progress is the beginning of the effect, 100% is a constant jittering of all the characters.
- **Progress Type**
 - **Absolute**: 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative**: 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction**, sets the direction of the text effect sequence, you can choose between the following options:
 - **Left**: Starts with the first character.
 - **Right**: Starts with the last character
 - **Random**: Uses a random order.
 - **Static**: All characters are processed at the same time.
 - **Wave**: Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center**: Starts the effect from the center of the text.
 - **2 Center**: Starts the effect at the same time from the beginning and the end of the text. They meet at the center.
- **Interpolate**, choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range**, defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %**. This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed**, specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.
- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.

- **Use Axis**, defines on which axis or axes the characters moves to create the jittering effect.
- **Range**, sets the range of the jittering movement.
- **Randomness**, sets the intensity of the jittering movement.
- **Lock Axis**, with this option enabled, the "Range" and "Randomness" will be the same for all axes. If you disable it, Range- and Randomness parameters for each of the axes will be displayed and you can set them individually.

TfxJitterScale



TfxJitterScale

Parameters

- **Progress %**, 0% progress is the beginning of the effect, 100% is a constant jittering of all the characters.
- **Progress Type**
 - **Absolute**: 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative**: 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction**, sets the direction of the text effect sequence, you can choose between the following options:
 - **Left**: Starts with the first character.
 - **Right**: Starts with the last character
 - **Random**: Uses a random order.
 - **Static**: All characters are processed at the same time.
 - **Wave**: Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center**: Starts the effect from the center of the text.
 - **2 Center**: Starts the effect at the same time from the beginning and the end of the text. They meet at the center.
- **Interpolate**, choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range**, defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %**. This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed**, specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.

- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Anchor horizontal**, sets the anchor point for the characters on the horizontal plane.
- **Anchor vertical**, sets the anchor point for the characters on the vertical plane
- **Use Axis**, defines on which axis or axes the characters scale to create the jittering effect.
- **Min**, sets the minimum scaling for the characters.
- **Max**, sets the maximum scaling for the characters.
- **Randomness**, sets the intensity of the jittering movement.
- **Lock Axis**, with this option enabled, all the axes get the same "Min.", "Max" and "Randomness" settings. If you disable it, these parameters will be visible for each of the axes and must be set individually.

TfxRotate



TfxRotate general description

The TfxRotate function allows you to create an effect where the characters of a text rotate on the X- Y- or Z-axis. To use the function, add it onto a container which holds a text. To create a backside for the characters, add the Expert function and enable the "Show backside" option.

Parameters

- **Progress%**, 0% progress is the beginning of the effect, 100% the end. Animate this value from 0% to 100% to see the effect or from 100% to 0% to animate the effect backwards.
- **Progress Type**
 - **Absolute**: 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative**: 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction**, sets the direction of the text effect sequence, you can choose between the following options:
 - **Left**: Starts with the first character.
 - **Right**: Starts with the last character
 - **Random**: Uses a random order.
 - **Static**: All characters are processed at the same time.
 - **Wave**: Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center**: Starts the effect from the center of the text.
 - **2 Center**: Starts the effect at the same time from the beginning and the end of the text. They meet at the center.
- **Interpolate**, choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range**, defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.

- **Wave Center Size %.** This parameter is only visible when “Wave” is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed,** specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.
- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Anchor horizontal,** sets the anchor point for the characters on the horizontal plane.
- **Anchor vertical,** sets the anchor point for the characters on the vertical plane
- **Use Axis,** defines on which axis the characters rotate.
- **Begin,** sets the initial rotated position of the characters.
- **End,** sets the ending rotated position of the characters.

TfxScale



TfxScale general description

The TfxScale function allows you to create a scaling animation of the characters. To use the function, drag it onto a container which holds a font. To create an animation, animate the “Progress%” value.

Parameters

- **Progress%,** 0% progress is the beginning of the effect, 100% the end. Animate this value from 0% to 100% to see the effect or from 100% to 0% to animate the effect backwards.
- **Progress Type**
 - **Absolute:** 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative:** 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each character).
- **Direction,** sets the direction of the text effect sequence, you can choose between the following options:
 - **Left:** Starts with the first character.
 - **Right:** Starts with the last character
 - **Random:** Uses a random order.
 - **Static:** All characters are processed at the same time.
 - **Wave:** Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center:** Starts the effect from the center of the text.
 - **2 Center:** Starts the effect at the same time from the beginning and the end of the text. They meet at the center.

- **Interpolate**, choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range**, defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %**. This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed**, specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.
- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Anchor horizontal**, sets the anchor point for the characters on the horizontal plane.
- **Anchor vertical**, sets the anchor point for the characters on the vertical plane
- **Use Axis**, defines on which axis or axes the characters scale.
- **Begin**, sets the initial size of the characters.
- **End**, sets the ending size of the characters.
- **Lock Axis**, with this option enables the "Begin"- and "End" scaling are the same for all the axes. If you disable the parameter, "Begin" and "End" must be set for all the axes individually.

TfxWrite



TfxWrite general description

The TfxWrite function allows you to create an effect where text characters get displayed as if they were written by the keyboard in real time, a sort of "Type-writer effect". To use the function drag in onto a container that holds a font. The text field in the font text editor does not need to contain any text, since the text for this plug-in is entered in the plug-in's editor. Type the text you want to be written in the text field in the plug-in editor. To create an animation, animate the "Progress %" value.

TfxWrite parameters

- **Progress%**, 0% progress is the beginning of the effect, 100% the end. Animate this value from 0% to 100% to see the effect or from 100% to 0% to animate the effect backwards.
- **Progress Type**
 - **Absolute**: 100% progress will animate all characters of a text, regardless how many characters it has.
 - **Relative**: 100% progress animates 10 characters. This is needed to adjust the timing of several text objects with different sizes. The effect speed should be e.g. 5 characters per second, so the animation must

be from 0% to 100% in two seconds. This will work for text with 10 characters or less. If you want to use longer texts animate the progress value over 100% (10% for each characters).

- **Direction**, sets the direction of the text effect sequence, you can choose between the following options:
 - **Left**: Starts with the first character.
 - **Right**: Starts with the last character
 - **Random**: Uses a random order.
 - **Static**: All characters are processed at the same time.
 - **Wave**: Starts with the first character, animates the effect from 0-100% and the down again to 0%.
 - **Center**: Starts the effect from the center of the text.
 - **2 Center**: Starts the effect at the same time from the beginning and the end of the text. They meet at the center.
- **Interpolate**, choose between a soft or a linear interpolation of the transition from character to character.
- **Effect Range**, defines how many characters that are to be in the effect processing at the same time. If for instance the Effect Range is set to 4, and you manually increase the progress value, you will see that when the fifth character starts the process, the first is finished, when the sixth starts, the second is finished, and so on.
- **Wave Center Size %**. This parameter is only visible when "Wave" is chosen as the effect direction. Its effect is only visible when the effect range is set relatively high. It defines how quickly the effect is to reach the top of the wave sequence. If for instance the Effect Range is set to 10 and the Wave Center Size % is set to 80, all characters except the one at each end will be at the top of the wave sequence. The one at the left end will be on its way up and the one on the right end will be on its way down.
- **Random Seed**, specifies a seed for the random number generator when a random direction is chosen. Even though the program uses random numbers, the animation for a specific random seed will always look the same. This is typically useful if you combine two different text effects.
- **Ignore Space** - Ignores Spaces when the effect range is calculated. If you use Mulibyte fonts, you should uncheck this option.
- **Ignore Newline** - Ignore newlines when the effect range is calculated. If you use multibyte fonts, you should uncheck this option.
- **Right to Left** - Makes the effect progress from right to left. If you are using a right to left language, you should check this option.
- **Fill with Space**, with this option enabled, the function will add a space for every character not jet written. The bounding box of the text is then sized correctly from the start. If you use an Auto Follow function connected to the text, enabling this option will create a better result.
- **Effect**, allows you to select an effect to be executed as each character gets written on the screen. There are four effects to choose from and they have in addition their own parameters.
 - **Pop**, is the basic effect where the characters just pop up on the screen without any other sign being displayed.
 - **Count**, creates a counting sequence by showing a random symbol before showing the character to be displayed.
 - **Random**, This (in combination with effect range) writes random characters before the final character is displayed. Use a non proportional font when you use this function to get the better looking effect.
 - **Cursor**, displays a cursor that writes the characters. When you select this option, a text field opens below where you can choose which symbol to use as cursor.
- **Effect Speed**, is an additional parameter that is visible if you have selected "Count" or "Random". It sets the speed of the effect that shows the preceding symbol.
- **Selection**, is an additional parameter to the "Random" effect. It allows you to select between some preset character ranges to use for the effect.
- **Cursor**, allows you to enter the character which is to be used as a cursor.

Advanced counter



Advanced Counter general description

The Advanced Counter function allows you to easily create an animated counting sequence. It shows values with decimals and it has the possibility to create a user defined format mask, and it can have prefixed values. To use the function, add it to a container holding a font and animate the value parameter.

Advanced Counter parameters

- **Value**, sets the current value of the counter. Animate this value to create the counting sequence.
- **Leading blanks**, you can here choose what the counter is to do with leading blanks. They can either be cut away or put in as blanks, star-symbols, number signs or punctuation marks.
- **Force Prefix**, by enabling this option, both positive and negative value will have a prefix, not only negative values as is the default.
- **Delete unnecessary commas**, removes superfluous commas when a specified format mask has more commas than the number needs. If you for instance have defined the mask: "\$###,###,###,###.##" and you enter the number 4120.37, it will be displayed as "\$,,4,120.37". If you enable this option, the number will get a correct format: "\$4,120.37"
- **Format mask**, allows you to define a format mask. Each number sign symbolize an element of the total number. To alter, add or remove number signs. You can also add constant values, e.g. DM, NOK

Auto rotate



General description

With the Auto rotate function, a container can be set to rotate round one of its axes continuously.

Parameters

- **Axis**, selects the rotation axis, X, Y or Z.
- **Direction**, selects the direction of the rotation, right or left.
- **Velocity [°/Frame]**, sets the speed of the rotation. The unit is "Degrees per frame".
- **Ease In [Frames]**, sets the number of frames the object will use to accelerate smoothly from no motion to the speed set in the Velocity parameter.
- **Ease Out [Frames]**, sets the number of frames the object will use to retard from the rotation speed to a stop.
- **Starting Angle[°]**, sets the angle from the object's current position, from which the object is to start from when the "Start" button is pressed.
- **Ping Pong Mode**, enables a "ping pong" motion where the object rotates forwards and backwards between two angles on the axis.
- **PP Angle 1 [°]**, sets angle 1 for the "Ping Pong Mode".
- **PP Angle 2 [°]**, sets angle 2 for the "Ping Pong Mode".
- **Start Button**, initializes the rotation from the starting angle selected and with the "Ease in" value selected.
- **Stop Button**, stops the rotation. The stop motion is influenced by the "Ease out" parameter.
- **Run Button**, starts the rotation from the point where it was halted. The value of "Ease in" is disregarded.

- **Halt Button**, stops the rotation without regard to the “Ease out” parameter.

Colorize



General description

The colorize function allows you to transform the colors of containers within a container group. If you for instance have 15 subcontainers in a group, the colorize function assign numbers each container by the order they have in the group. You can then define for each subcontainer which color it should have to start with and with color it should change to. The function can define up to 10 colors. Each color is able to influent more than one container. The “num color” parameter allows you to link multiple containers to one color in the colorize function. Each of the subcontainers must have its own material for the function to work.

Parameters

- **Colorize**, runs the color transformation. One number corresponds to one container.
- **Animation Start**, allows you to select the starting point for the color change.
- **Default Color**, sets the color for containers in the group who are not selected for color transformation. To set it, use the color editor below or drag a material from the material database and onto the small square.
- **Numb Color 1A-10A**, assigns the starting colors to containers, from which the color change starts. The value sets the number of containers to be linked with the corresponding color in the “Color 1A-10A” parameters. The function starts counting from the first “undefined” container. This means that, if color 1A have the value 2, color 2A have the value 3 and 3A have the value 1, the two first containers gets the properties of color 1A, container 3-5 get the properties of color 2A and container 6 gets the properties for color 3A.
- **Color 1A-10A**, sets the colors for 1A-10A. You can set each color using two methods:
 - Either select the color by clicking the color icon and then set the color properties at the color editor at the bottom, or
 - Drag a material icon from the material database onto the color icon of the color you want to set.
- **Num Color 1B-10B**, assigns the ending color to containers, to which the color change ends. The value sets the number of containers to be linked with the corresponding color in the “Color 1B-10B” parameters. The function starts counting from the first “undefined” container. This means that, if color 1B have the value 2, color 2B have the value 3 and 3B have the value 1, the two first containers gets the properties of color 1A, container 3-5 get the properties of color 2B and container 6 gets the properties for color 3B.
- **Color 1B-10B**, sets the color for 1B-10B. To select colors see parameter “Color 1A-10A”.
- **Rebuild**. After having made changes to colors and assigning containers, click rebuild to apply the changes.

Counter



Counter general description

The counter is a simple function that can create a count up or count down at so to say any given range and at any given speed. It shows integral numbers. To use the function create a container with the wanted font and material and add the Counter function.

Counter parameters

- **Number**, sets the current number of the counter. Animate this value to create the wanted count-up or -down.

LOD



LOD general description

LOD is a mechanism for controlling the level of complexity of a 3D object, depending on camera position and object size. As an object becomes smaller on the screen, the object's polygons become smaller. At certain size, those polygons are small enough to be replaced by a lower number of larger polygons, without losing realism. The goal is to maintain the overall shape of the object so it will not change significantly, when the number of polygons is changing.

Viz|Artist has a built-in LOD mechanism for text objects and internal objects, such as Sphere, Cylinder etc. This mechanism changes the tessellation of the object, based on its size on screen. However, there is no LOD mechanism for imported objects and models. This is where the LOD plug-in comes in handy. The LOD plug-in enables switching between different representations (levels) of the same object, based on its distance from the camera and on the zoom level of the camera. A switching range is assigned to each representation - to determining the specific distance from camera in which it will be replaced by the next representation.

In addition to that, the zoom of the camera is taken into consideration; objects that are far away from the camera will look closer (and larger) at a narrow zoom position, or smaller at wide zoom. To accommodate for different zoom values, the LOD plug-in assumes that the switching ranges are relative to a reference zoom value. When the actual zoom value differs from the reference zoom value, the switching ranges are automatically adjusted.

How to set up the LOD plug-in

In order to use the LOD plug-in on an object, one has to import several representations of that object from an external source. The different representations of the object have to be placed under a group node, in the order of detail, i.e. the most detailed level will be on the top of the group. When the LOD plug-in is placed on the group container it will make only one of its children (levels) visible at a given time, based on the distance between the center of the object and the camera, as well as the current zoom value, and other selectable parameters of the plug-in.



LOD plug-in parameters

- **Lock** - In certain cases it is desirable to disable the distance switching and lock the LOD plug-in to show one of the levels.
- **Lock On Index** - The index of the selected level when Lock mode is on.
- **Center** - Defines whether the distance from the camera will be measured from the origin of the object (Automatic) or from a specific point in the object, e.g. one of the corners (Manual)
- **Center X,Y,Z** - The point from which the distance from the camera is measured (in object coordinates), if the Center parameter is set to Manual.
- **Range Scale** - A scale factor on the switching ranges. This scale factor allows shortening or lengthening the distance at which object representations are switched.
- **Range 0..n** - the switching ranges. Those are the distances at which object representations are switched. The number of entries is n+1 where n is the number of children in the group on which the LOD plug-in is assigned
- **Initialize** - pressing the initialize button will re-initialize the LOD plug-in. This is required if the number of children of the LOD group has changed after the LOD plug-in is assigned to the group.

The LODManager plug-in



LODManager is a scene plug-in that controls all the LOD plug-ins in the scene. The LODManager plug-in is added automatically to the scene when the first LOD plug-in is placed on a group.

LODManager plug-in parameters

- **Reference Zoom** - This zoom value is the reference zoom that is used to correlate switching ranges based on the actual camera zoom value.
- **Reference Width, Height** - currently unused.
- **Range Scale** - this is a global range scale, used to scale all LOD ranges on all LOD plug-ins in the scene.

Magnify



General description

The Magnifying function offers the possibility to create a simulation of a looking glass and other similar effects. At an area defined by an overlying object an image can be shown in a magnified way. To set up the function you first create the container which is to be magnified. This container must contain an image or texture. The object you want to use as magnifier must then be added as a subcontainer to the container who is to be the target of the magnifying. You add the same image to the subcontainer and thereafter add the magnifying function. Remember to move the subcontainer a little bit nearer the camera than the parent container so they don't have the same Z-value.

Parameters and functions

- **Scale**, sets the scaling of the magnifying effect
- **Effect Type**, adds an extra effect, Blurring or Pixelation. This only works on Onux.
- **Effect Value**, sets the value for the selected extra effect.
- **Alpha**, sets the alpha value for the container
- **Rebuild**, after having made changes to texture, click rebuild to apply the changes.

Omo



Omo general description

The omo plug-in gives the user the possibility to create a very realistic animation of a complex imported 3D object, where the object not only changes position or rotates, but where the it changes form and moves in a realistic way, like a man walking or a fish swimming. The function can also create animation sequences with other sorts of objects. Anything you can put in a container can be used as an element in an Omo animation process. The Omo function is to be placed on a group container. What the function does, is to hide all the subcontainers and show them one by one. Which one who is shown, is controlled by the "Visible Container" parameter. By animating this value an animation sequence of all the containers is made. The function can also be used to show one by one of many containers.

Omo parameters

- **Visible Container**, allows you to select which container who is to be visible.
- **Keep visible**. With this option selected, all containers are kept visible after they have been revealed one by one.
- **Initialize**. Click this button to initialize the Omo function. All containers in the group is then hidden except the one selected in the "Keep visible" field.

Parlament



General description

The Parlament function is specially developed for creating visualizations of parliament seats in graphics designed for election programs, but it can of course be used in other ways. The function creates a "parlament like" seating structure using an object of your selection. Add a graphical object to the transformation tree, set the size, add a material, rotate and set other parame-

ters. Drag the Parliament function onto the container. By default the seating is created with 40 seats in 3 rows. The function creates then by default 40 subcontainers that are copies of the core-object. All the newly created subcontainers have the same properties as the source object except from the position. This is set by the function to create the parliament structure. Take notice! If the object you use for creating the seats contains many polygons, the total product of the parliament container can quickly be very heavy to render.

Parameters and functions

- **Number of elements**, allows you to set the number seats to be created in the parliament structure.
- **Number of rows**, allows you to decide how many rows the parliament seats should be distributed over.
- **Rebuild Geometry**. After having made changes to either "Number of elements" or "Number of rows", you must click on this button to apply the changes.
- **Inner Radius**, sets the inner radius of the parliament.
- **Outer Radius**, sets the outer radius.
- **Start Angle**, sets the starting angle which by default is - 90° from the Y-axis.
- **End Angle**, sets the ending angle, default 90°.

Rotation Order



General description

RotationOrder changes the rotation order of a container in Viz to match the XSI rotation order. This plug-in is automatically added to some containers when XSI scenes are imported into Viz.

Slide Show



Slide Show general description

The Slide Show function allows you to create a sequence where a group of containers fades in and out one by one in the same manner as normal slide show. To use the function create a group and add the subcontainers you want to be shown as slides. You can position them the way you want in advance, or you can use the offset function to create a moving sequence. Add the Slide Show function to the group container and animate the blend value to create the slide show sequence. When the function is added and initialized, each of the subcontainers automatically get an alpha icon.

Parameters

- **Blend value**, controls the slide show sequence. Animate this value to create the change from slide to slide.
- **Use offset**, enables this option if the slides have the same initial position and you want them to move along some of the axes as the slides are being shown.
- **X, Y and Z offset**, sets the offset values of each slide calculated from the one previously shown.

Soft Clip (Texture)



The Soft Clip plugins

Soft clip is an viz plugin that can show AVI clips, either projected on a texture, or rendered directly.

The softclip plugin comes in two variants:

1. SoftclipDrawPixels, which is a geometry plugin and draws pixels directly into the frame buffer.
2. SoftclipTexture, which projects the video on a texture.

Soft Clip general description

Choose a video file to play. Playback can be achieved by either animating the frame numbers or by turning on the Live Playback. Since the plugin projects the video onto a texture, the video can be mapped onto an object just as with an normal texture. Considerations regarding codecs, performance and other important things to consider is the same for this plugin as for the Draw pixels variant, see ["Soft Clip Draw Pixels" on page 124.](#)

Soft Clip parameters

- **Clip File** - The video file to play
- **Live Play** - in live playback mode, the frames will be displayed synchronized to viz rendering, or according to the Frames Per Second (FPS) setting, depending on the play mode that is chosen.
- **Frame number** - This is visible if Live Play is disabled. Animate this value to create a video playback.
- **Play mode**
 - **Synchronized** - Will synchronize the frame rate of the AVI to the output framerate of Viz|Engine, i.e. for each rendered viz frame, the video will advance exactly one frame.
 - **Frames per second** - In FPS playing mode the plugin will control frame rate by correlating the value to the rendering frame rate. This will only work if rendering rate is synchronized with video output.
- **Frames per second** - Sets the frame rate.
- **Loop video** - Loops the video
- **Use decode thread** - Choose whether to use threading for video-decoding. On multi-processor machines this can gain you performance, but threading makes scheduling less predictable and may cause viz frame rate to be less stable on a single CPU machine.
- **Preload frame count** - Sets the number of frames that should be pre-loaded when the
- **Frame skipping** - When on, viz will never wait for decoded frames, i.e. video frames will be skipped rather than forcing viz to drop or delay frame rendering.
- **Force Opaque** - When showing videos where the codec leaves the alpha channel blank, the video rendering may become completely transparent. This option will fill the alpha channel, thus making the video completely opaque.
- **Preferred CPU** - If you have more than one CPU, you can choose which one the decoding process should use: 0=any
- **Priority** - Sets priority for the decoding process.
- **Auto scale texture** - Scales the texture to fit

Temo**General description**

The Temo plug-in gives the user the possibility to create an image animation sequence, using a method much like the one used for making a cartoon film. The basic input for the function is a single image consisting of many tiled equally sized squares set up in a matrix. Each square of the image is made up to be a snapshot of an animation sequence, just like each picture frame is on a normal movie film. The image must be made up in advance with the aid of a picture editing program e.g. "Photoshop". In the parameter editor you tell the Temo function how many tiled squares there are in the X- and the Y-axis. The function then is able to show the tiles of the image one by one. By animating this, a "film like" sequence is created. The plug-in has only a small influence on the rendering performance.

Parameters:

- **Number of tiles horizontal.** Here you define the number of tiles in the X-axis of the source image.
- **Number of tiles vertical,** the same as above but for the Y-axis.
- **Show tile number,** selects which tile who is to be shown. The range of tile numbers is the product of the two above parameters. If there is 3 in the horizontal and 4 in the vertical plane, the total is of course 12. The tile with the coordinates X1Y1 is shown first, then X2Y1 and so on. By animating the whole range of this value, a "film animation" can be created. The quality of the animation depends of course on the number of tiles and on the degree of transformation from one tile to the next throughout the whole sequence.

Text Link**Text Link general description**

The Text link function copies the text from the source container to 1 to 10 linked containers. The text you enter in the source container will be repeated in all the linked containers. The linked containers must contain a font. To use the function drag it from the function database and onto the container you want to use as text source, drag the containers you want to link onto a dropzone each in the plug-in editor. To apply changes press the initialize button.

Text Link parameters

- **Container 1-10,** are the dropzones for the containers you want to link.

Image Link**Image link general description**

The Image link function copies the image/texture from the source container to 1 to 10 linked containers. The image you load in the source container will be repeated in all the linked containers. To use the function drag it from the function database and onto the container you want to use as image source, drag

the containers you want to link onto a dropzone each in the plug-in editor. To apply changes press the initialize button.

Image link description

- **Container 1-10**, are the dropzones for the containers you want to link.

TextParam



General description

The TextParam function gives the user the possibility to animate text parameters. The standard text parameters in the Text Editor can not be animated. Drag the TextParam function onto a container with a text element, click on the TextParam icon and animate the desired values. The parameters can only be used for animation. The settings are not visible on the text until an animation update has been made.

Parameters to be animated

- **Use Kerning**, enables/disables use of Kerning
- **Kerning**
- **Line Spacing**
- **Word Spacing**
- **Use Soft Shadow**. To use Soft shadow, select the level you want. Level 4 is the most blurred/softest shadow. The drop shadow option must be enabled for the soft shadow to be visible.
- **Use Drop Shadow**, enables/disables drop shadow.
- **Direction**, sets the direction of the drop shadow
- **Distance**, sets the distance of the drop shadow.
- **Z-Offset**, sets the offset of the drop shadow in the Z-axis.
- **Color**
- **Alpha**

Shader Function plug-ins

Effect folder **Frame Mask**



The Frame Mask plug-in is a rectangular soft mask. Add the plug-in to the container that you want to mask.

Frame Mask parameters

- Left/Right/Top/Bottom Edge position - sets position for the edges of the mask
- Left/Right/Top/Bottom Gradient - sets the width of the Soft gradient at the borders of the mask.

Image Mask



The Image Mask Plug-in is a mask that takes its form from an Alpha image. Add the plug-in to the container that you want to mask and add an image to the plugin editor that gives you the desired form for the mask

Image Mask Parameters

- Image - Sets the alpha image that dictates the form for the mask
- Blend - Sets the blending between the mask and the masked object
- Position X - Sets the X-position of the mask texture
- Position Y - Sets the Y-position of the mask texture
- Scale X - Sets the X-Scale for the mask texture
- Scale Y - Sets the Y-Scale for the mask texture
- Wrap - Choose whether to Clamp or repeat texture.
- Mask Aspect - Sets size and aspect to screen or to the original aspect of the image used as mask.
- Invert - Inverts the mask

Soft Mask



The Soft Mask Plug-in is Soft wipe mask with a progress value to animate. The progress value moves the wipe across the screen. Add the plug-in to the container that you want to mask.

Soft Mask Parameters

- Progress - Moves the mask across the screen
- Rotation - Rotates the mask
- Center X - Sets X center point
- Center Y - Sets Y center point
- Gradient Width - sets the width of the soft alpha gradient
- Invert - Inverts the mask
- Mirror Gradient - Mirrors the gradient
- Blank - Offsets the gradient starting point

Water



This plugin simulates the surface of water. It is currently in an experimental state.

Filter folder **Blend Image**



The Blend Image plug-in lets you blend two images in a number of ways. Add the plugin to the image container, add an image to blend with in the plug-in editor and choose blend mode. Adjust the other parameters to obtain the desired effect.

Blend Image parameters

- Image - Drag and drop the image you want to blend with the container image.

- **Mode** - Choose which blend mode to use.
- **Blend** - Sets the blending factor.
- **Position X** - Positions the blend image on the x-axis
- **Position Y** - Positions the blend image on the y-axis
- **Scale X** - Scales the blend image in X
- **Scale Y** - Scales the blend image in Y
- **Wrap** - If the blend image is smaller than the container image, select if it should be repeated or clamped.

Blur



The blur plugin lets you blur an image or a texture.

Blur parameters

- **Quality** - Set the quality for the blurring, the higher quality the more performance is used.
- **Range** - Set the degree of blurring.

Color Correction



The Color Correction plug-in gives you access to standard adjustment properties such as:

- Brightness
- Contrast
- Saturation
- Hue

Radial Blur



Radial blur tries to simulate the blur you would get through a zooming or rotating camera.

Radial Blur parameters

- **Quality** - Sets the smoothness of the blur.
- **Center X** - Sets X position for the origin of the blur.
- **Center Y** - Sets Y position for the origin of the blur.
- **Inner Range** - Defines how much of the image that will be affected by the blurring.
- **Scale** - Applies the effect: 0 = no effect, 100 = full effect.

Sepia



Sepia tone is a type of monochrome photographic image in which the picture appears in shades of brown as opposed to grayscale as in a black-and-white image. Opposed to a standard Sepia effect that interpolates from grayscale to brown, this plugin can interpolate between any color you like.

Sepia parameters

- **Desaturation** - sets the color intensity.
- **Tone** - Sets the amount of sepia tone
- **Light Color** - Sets color and alpha for the light color component that the sepia tone is made from.
- **Dark Color** - Sets color and alpha for the dark color component that the sepia tone is made from.

Sharpen



The Sharpen plug-in focus blurry images by increasing the contrast of neighboring pixels.

Sharpen parameters

- **Quality** - sets the quality of the sharpen effect. High quality gives the best visual result but cost the most performance.
- **Range** - Sets how intense the effect should be.
- **Scale** - Applies the effect: 0 = no effect, 100 = full effect.

Material folder **Anisotropic Light**



Anisotropic Light is a per pixel lighting model that assumes that the surface has microscopic facets that are uniformly distributed in any direction on the surface. Anisotropic lightning is meant to simulate the effects of lightening on a grooved surface like a record, human hair, brushed metal etc.

All the shader properties are defined by the material on the container. Only light source number 1 is used and the light is always treated as a local light source.

Cell Shader



The Cell Shader plug-in creates a stepwise color shading that can be used to make the object appear like a cartoon drawing.

Cell Shader parameters

- **Shades** - Sets the number of shades.
- **Edge Width** - Sets the width of the shade border
- **Surface High** - Sets the color for the surface that is nearest the light source.
- **Surface Low** - Sets the color for the surface that is furthest away from the light source.

Gooch



Gooch shading is a form a shading suitable for depictions of metallic surfaces.

- **Surface** - sets the color and alpha for the “normal” surface color
- **Cool** - sets the color and alpha for the “Cool” surface, hence the surface furthest away from the light source.
- **Warm** - sets the color and alpha for the “Hot” surface, hence the surface nearest the light source.
- **Highlight** - set the level of highlight or shininess in the area closest to the light source.
- **Highligh Alpha** - sets the alpha for the highligh area
- **Texture Alpha** - sets the alpha for the texture on the object.

Texture folder **Drop Shadow**



The drop shadow plugin makes a texture throw a static shadow.

- **Distance** - Sets the distance between the texture and the shadow.
- **Direction** - Sets the direction from the texture to the shadow
- **Threshold** - Sets the spacing between the texture and the shadow when the texture is overlaying the shadow.
- **Color** - sets the color for the shadow
- **Alpha** - sets the transparency for the shadow.

Emboss



The Emboss plug-in Makes a selection appear raised or stamped by converting its fill color to gray and tracing the edges with the original fill color.

Emboss parameters

- **Direction** - Sets the direction for the emboss effect
- **Emboss** - Sets the level of emboss effect

Animation

Create an Animation

Animation menu



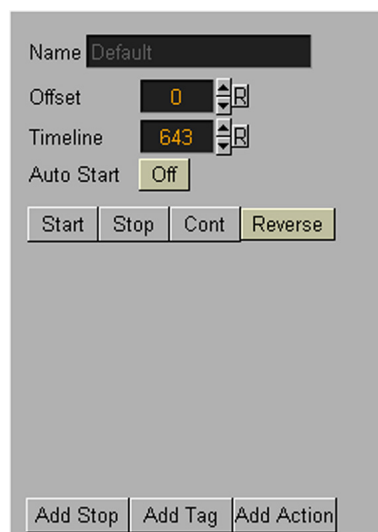
Animations are created and controlled using the buttons above the render window: From left to right the controls are:

- “Default” or another director name. This button changes its name depending on which director it enables. It can enable a specific director or all directors. To the right of this button is a small button which you use to lock the director choice you have made.
- “Start”, starts an animation from the beginning.
- “Stop”, stops a running animation.
- “Cont”, continues an animation from the point it was stopped by either the stop button or a stop keyframe.

- "Prev", jumps to the previous keyframe of the selected container
- "-", goes back 10 fields
- The value field shows the current time line position
- The up/down arrows jump forward and backward by one field
- "+", jumps forward by 10 fields
- "Next", jumps to the next keyframe of the selected container.
- "Update", creates a new keyframe for the selected object.
- The input field to the right of the 'Update' button allows you to choose the number of fields the time line will jump ahead when creating a new keyframe.
- "R", to the outer right allows you to reverse an animation sequence. With that enabled an animation will be run backwards when the start button is clicked.

Director editor

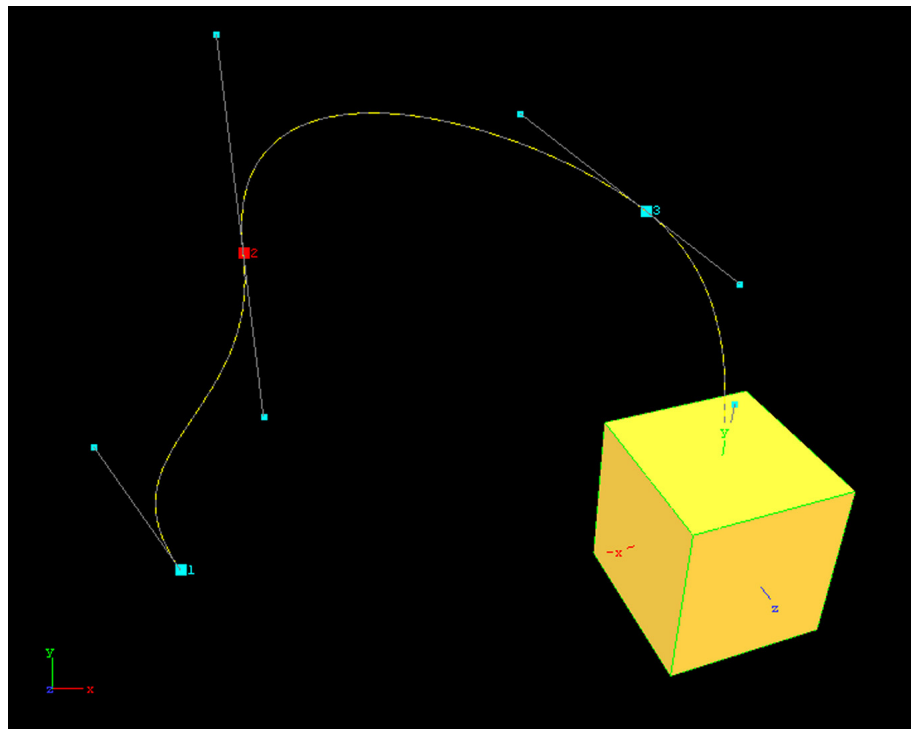
The director editor opens to the left of the stage when you click on a director.



It has the following parameters/functions:

- **Name** - Sets the name of the director. (Default director is always default and can not be changed)
- **Offset** - Sets an offset for the director on all its animation channels
- **Timeline** - Shows the current position on the directors timeline
- **Auto Start** - With this option enabled, the director will start automatically after the scene has loaded.
- **Start/Stop/Cont/Reverse** - Control buttons to start, stop and continue the director. Reverse sets the director to run backwards.
- **Add Stop** - Adds a stop point on the timeline
- **Add Tag** - Adds a tag on the timeline
- **Add Action** - Adds an action channel.

Basic animation steps

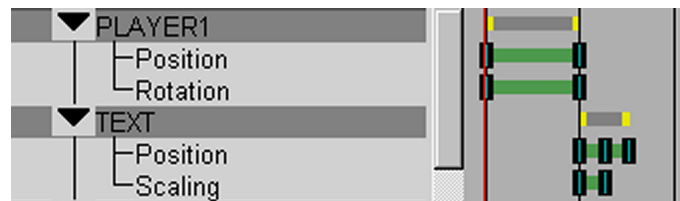


In this example an objects position is animated, but you can of course animate many more properties of an object. To create a basic animation, do the following:

- Position an object somewhere in the rendering area
- Make sure the object is selected. You should see the green bounding box. If you don't see it, click once on the object and make sure that the bounding box button ('BB') to the left of the render window is switched on
- Make sure the timeline value is zero and press update once. This tells the program to remember the properties of the object at this position. The timeline value has increased with 50, (it will differ if you have set the "framerate per second" to something else).
- Move the object to another position, either with the mouse or by altering the values in the transformation editor, and press update again. The object's container in the transformation tree now shows an animation icon and the default director in the stage editor lists the object and its position channel. The program has now created two keyframes, one which describes the properties of the object at the starting point, and one describing the endpoint properties. You can open the stage and click on the keyframes at the objects position channel and see their properties in the keyframe editor to the upper right.
- Press the start button to run the animation.
- To add an additional keyframe, enter 100 in the time line value field, move the object to a new place and press update. A third keyframe have now been added. To see the path of the objects movement between the keyframe points, right click on the object and select animation from the menu.
- The process is the same for other properties than position, enter value of the first keyframe and press update, alter the desired properties an press update again. This is in principle the method for creating an animation.

Editing an Animation

Editing animations in the Stage Editor



Adding Directors

To add a director, click the "Add Director" button and enter a name for the new director. You can move objects from the default director onto the new one. New objects being animated are always created in the default director. If you wish to locate an animated object under another director, it must be moved to another director after the animation sequence is created.

Deleting Directors

To delete a director, click on the director symbol in the stage three, hold the button down and drag it onto the recycle bin

Run a single director

To run the animations of a single director, press the small button to the left of the start button in the rendering window. The "director" button is then enabled, and you can press it to choose the desired director. When the director is chosen, press play, to run the animations located in that director. You can also run a single director by selecting it in the stage tree and use the buttons that appear in the director editor to the left. If you have selected a specific director and thereafter edits an object in the render window that is located under another director, the "director button" changes its state to showing that director instead. To be able to edit other objects without changing the director selection, you must lock you selection with the small button to the side of the director button. With that button in a disabled state, the director selection will not change as you edit object under other directors.

Adding keyframes

To add a new keyframe can be done in two ways.

1. Give the object the wanted properties, and press the update button above the rendering window. This will create a new keyframe at the end of the animation channel.
2. Give the object the wanted properties and click the add keyframe button in the stage editor. Where on the animation channel the new keyframe is created, depends on the position on the thin read timeline in the stage.

Either way the new keyframe contains the current animation channel properties for the object at that given time. If you set the thin red time control line on a time point outside the current animation sequence without changing any properties for the object and then click the "Add keyframe" button, the new keyframe inherits the properties of the last or the first keyframe, depending on where you have placed the control line, in front or after the current sequence. Keyframes can be moved afterwards by either dragging them with the mouse, or by altering the "Time value" in the parameter section of the actual keyframe.

Deleting keyframes

To delete a keyframe, select it and press delete on the keyboard.

Adding stops

Stops can be added to a director's timeline. At a stop-keyframe, the animation sequence will halt until the "continue" button above the rendering window is pressed.

Adding actions

Actions can be added as sub elements to a director. A director can hold more than one action. To add an action channel, select the director you want to place it under and press the "Add action" button, on the prompt give the new action a name and press enter. A grey "action-bar" is created. The new action bar has no keyframes yet and therefore no timeline. Action keyframes must be added to the action. To add an action keyframe, select the action icon, set the thin red time control line at the desired moment on the timeline and click the "Add keyframe" button.

Adding Tags

You can add tags to a director. Select the director and press the add tag button. By giving the tag a name you can make an external control program or an action keyframe go to the point on the timeline marked by the tag. The syntax for an action keyframe command could be: THIS_SCENE*STAGE SHOW \$test. The tag name is in this example "test"

Action keyframes

The screenshot shows a dialog box for editing an action keyframe. It has the following fields and controls:

- Name:** My_action_keyframe
- Time:** 86
- Value:** RENDERER*STAGE*DIRECTOR*dir1*EVENT ADD 3.2 STOP
- Type:** Command (selected), Task
- Dir:** Both (selected), Normal, Reverse
- Execute:** [Execute]
- Navigation:** [Add] [Delete] [Prev] [Next]

The action keyframe editor has the following parameters:

- **Name** - Sets the name on the keyframe
- **Time** - Shows the keyframes time value
- **Value** - Enter you command or OS task here.
- **Type** - The action keyframes can initialize two main types of actions:
 - **Command** - is Viz|Artist internal commands. These can execute a great number of tasks in the program. Many elements can be automated by using commands. For documentation on this, "[External Control Commands](#)" on page 186. You can add multiple commands by separating them with the ';' semicolon character.
 - **Task** - is a normal shell command, like a you would write in a dos-prompt. Anything you can initialize from the OS command prompt can in principle be executed as a task in an action keyframe. If the path to the executable file contains spaces, you must set the complete path in quotation marks, like: "c:\program files\microsoft office\office\winword.exe".
- **Dir** - Sets which way, forwards or backwards, the animation should go to trigger the action keyframe.

- **Both** - The keyframe will be triggered when the animation goes forward and in reverse.
- **Normal** - The keyframe will be triggered when the animation goes forwards.
- **Reverse** - The keyframe will be triggered when the animation goes backwards.
- **Execute** - Executes the action
- **Add** - Adds a new keyframe
- **Delete** - Deletes the keyframe
- **Prev/Next** - Use these buttons to jump between the keyframes on the channel.

Editing keyframes

Keyframes can be edited in the keyframe editor. The editor of a keyframe opens in the left side of the stage editor, when the keyframe is selected. The type and number of parameters in the keyframe editor depends on the type of keyframe. For an overview of the range of parameters connected to keyframes, see ["Explanation of Keyframe parameters" starting on page 166](#).

Change a moving object's path between existing keyframes

There are two main ways to change the path of a moving object:

- Change the position of one or more of the keyframes. This can of course be done by altering the position values of each keyframe in the keyframe editor, but this is not a very intuitive method. The normal way to do this is to position the keyframes directly in the rendering window by using the mouse. Display the animation path by right clicking on the object and select "animation". With the left button pressed, change the path by dragging the keyframes to new positions.
- Change the interpolation between the keyframes by altering the weight and bias of the keyframe splines. This can be done in the keyframe editor under the submenu "path control" by altering the values of bias and weight and choosing between "smooth" and "linear" interpolation. The more intuitive method is to do this directly in the rendering window by changing the position of the keyframe spline handles with the mouse. Right click on the object and select "animation" to display the keyframes on the path and drag the keyframe splines around to obtain the desired form on the curve. Explanation of the "path control" parameters, see ["Special parameters on position keyframes" starting on page 166](#).

The path interpolation between all keyframes on a position path can be changed by choosing interpolation type on the parameter editor for the whole animation channel. You can choose between:

- Smooth interpolation
- Linear interpolation
- First/Last linear (Linear at the first and last keyframe and smooth between the rest)

More info on animation channels parameters, see ["Explanation of animation channel parameters" starting on page 165](#).

Change speed

To change the speed of a moving object between keyframes, alter the speed interpolation in the keyframe editor under the submenu "speed control". You can also set the whole animation channel, and thereby all keyframes, to linear or manual speed in parameter editor for the animation channel.

More info on the speed parameters for keyframes, see ["Special parameters on position keyframes" starting on page 166](#).

More on speed control for the animation channel see, ["Explanation of animation channel parameters" starting on page 165](#).

**Explanation of
animation
channel
parameters****Swing**

If the swing option is activated the animation channel goes back to the beginning keyframe when it reaches the end.

Loop

By enabling this parameter and selecting a value X in the field next to the button, the animation channel loops X times.

Interpolate Smooth

This parameter is first of all relevant to all sorts of position and movement channels, but it is also present on an alpha animation channel. By enabling this option the movement/transition is made using a smoothness interpolation function.

Interpolate Linear

This parameter is relevant to all sorts of position and movement channels, but is also present on an alpha animation channel. By enabling this option the movement/transition is made using a linear function.

First/Last Linear

This parameter sets the first and the last keyframe to a linear interpolation. The rest is spline controlled and a smoothness interpolation is used between the keyframes.

Linear Speed

This parameter locks the speed to a linear modulus. By default when you create an animation between three or more keyframes, the speed between each pair of keyframes is interpolated evenly on the distance between the keyframes. This means that if you have three keyframes, where the distance between the first and the middle is small and the distance between the middle and the last is big, the speed will be slower between the first and the middle compared with the speed between the middle and the last. If you define the speed for the animation channel to be linear, the program calculates the length of the whole animation spline curve and sets an even linear speed throughout the whole movement, removing differences related to difference in distance between keyframes.

Manual Speed

This option enables you to tweak the speed between the keyframes. By enabling this, the speed properties can be set individually on each keyframe using the left and right bias value and the left and right weight value.

Close Path

This option closes the path of an animation channel between the first and the last keyframe.

Axis Interpolation

This parameter is relevant to the rotation and animation channels. By enabling this the rotation channel is interpolated using an axis interpolation. With axis interpolation, rotations in the X, Y and Z axis are calculated individually. In many cases this will work fine, but in some situations the case can be that one axis gets rotated before the other instead of in a smooth manner. Some other strange effects can also occur. In these cases, the solution may be to select quaternion interpolation.

Quaternion Interpolation

This parameter is relevant to the rotation animation channel. By enabling this the rotation channel is interpolated using a quaternion interpolation. The quaternion interpolation is a complex model combining the rotation of the three axes producing a smooth rotation sequence. Each set of rotation values (x, y, z) is interpolated as position on a sphere. The quaternion animation animates along a path on the sphere

Explanation of Keyframe parameters

Parameters common to all keyframes

By clicking on a keyframe, the parameter editor opens in the left window in the Stage Editor. The number of parameters here varies depending on the types of keyframes you create. Some elements are however present on all types of keyframes

- **Name**, all keyframes can be given a name. This is not mandatory, but it is useful as an animation grows in size, and it must be done if you are going to call a keyframe by external control commands.
- **Time**, shows the frame number of the keyframe. To alter it, you can enter a new value directly into the field or you can drag the current keyframe along the time line at the stage.
- **Value**, shows the objects value or values. A color channel will have three values, R, G and B, a position also three, X, Y and Z. Keyframes which shows if a parameter is enabled or disabled have one value field showing the current status, like a hide/show keyframe.

Special parameters on position keyframes

Position keyframes have some parameters only relevant to the position animation channels:

- **Manual speed**, with manual speed selected, you can edit the frame time value in the field to the left, and thereby move the keyframe on the timeline. By increasing the value, the speed between the keyframe and the next will increase and the speed between the keyframe and the preceding keyframe will decrease and conversely if you decrease the value, the opposite situation is the case.
- **Path control**, smooth or linear, allows you to select how the path into and out from the keyframe is interpolated.
- **Path Smooth parameters**: These let you change the shape of the movement curve, by the means of left and right weight and bias. Left and right refer to the left or right of the current keyframe. The values can be changed in a looked manner where the right and the left value always changes the opposite way, in a mirrored way where both values changes equally, or in a single manner, where each value is changed individually.
 - **Bias** (left or right), represent the angle to the checkpoints to the left and right of the keyframe. The position curve between two keyframes is a Bezier curve with four checkpoints. In the rendering window you can visualize the curves by right clicking on the object and selecting animation from the menu. (The bounding box option must be enabled). The checkpoints to the left and the right of a keyframe are displayed with spline handles attaching the actual keyframe. By altering the value of the right and left bias, the angle of the handles change and the form of the path curve changes correspondingly. Most designers prefer to use the mouse with the left button pressed and drag the handles to the position that shapes the curve in the wanted shape.
 - **Weight** (left or right), represent the distance between the keyframe and the control points. The control point will have most influence on the curve at the nearest parts. By increasing or decreasing the weight, you can decide where on the curve will bend the most. As with bias,

this is normal done with aid of the mouse, dragging the checkpoint further away from the keyframe.

- **Speed control**, smooth or linear. This option allows you to select how the speed of the moving or transforming object is processed. Linear calculates the speed to the left and the right of the keyframe using a linear model. Smooth can be altered with the parameters below:
- **Speed Smooth parameters**
 - **Bias**. Even though it isn't visible, the speed between two keyframes is controlled by a Bezier spline curve, equal to the curve controlling the animation path. With that in mind it's easier to visualize how the speed is calculated. Bias in speed context is a value telling the render if a moving object should accelerate or retard and how much.
 - **Weight**, in the speed context, tells the render how far away from the keyframe the speed changes should inflict. If you have reduced the Bias by some value, the object slows down more than before when getting nearer the keyframe. By increasing the weight the slowdown takes place earlier.

Special parameters on rotation keyframes

Rotation can be interpolated in two ways:

- **Quaternion**, interpolate the rotation using a quaternion model. You can in addition select the number of spins the object should perform.
- **Axis**, smooth or linear. With axis interpolation you can choose between a smooth or linear interpolation.
- **Smooth parameters**
 - **Bias** (right and left), adjusts how much of the complete rotation that should be done nearer or further away from the keyframe.
 - **Weight** (right and left), adjust the range of the bias setting.

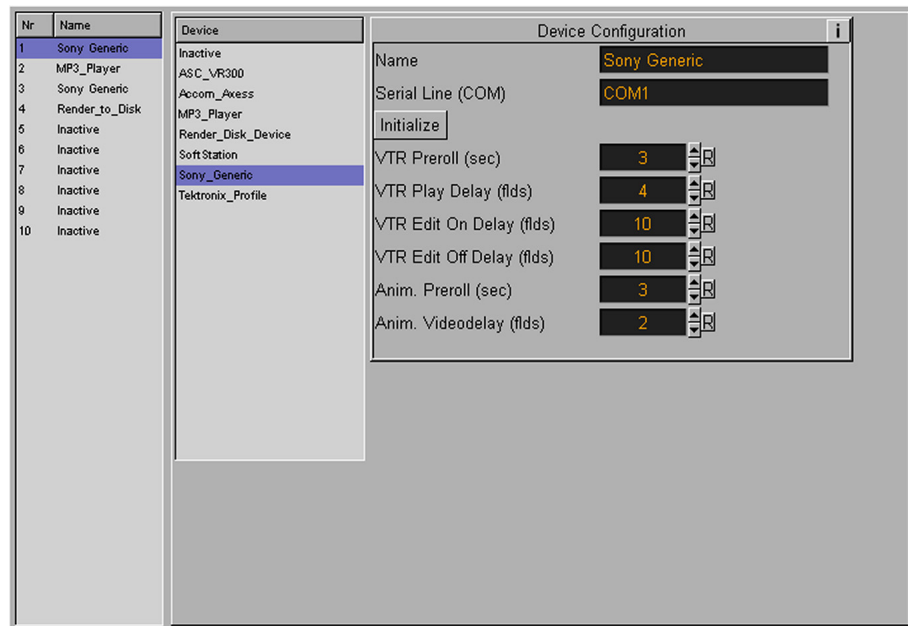
Special on "effect" keyframes

On some effect keyframes, such as alpha and shininess, you can choose if the transformation should be made in a smooth or linear manner. To the smooth option you can set weight and bias in the same manner as on speed keyframes.

Viz|Artist device plug-ins

Using the Device Manager

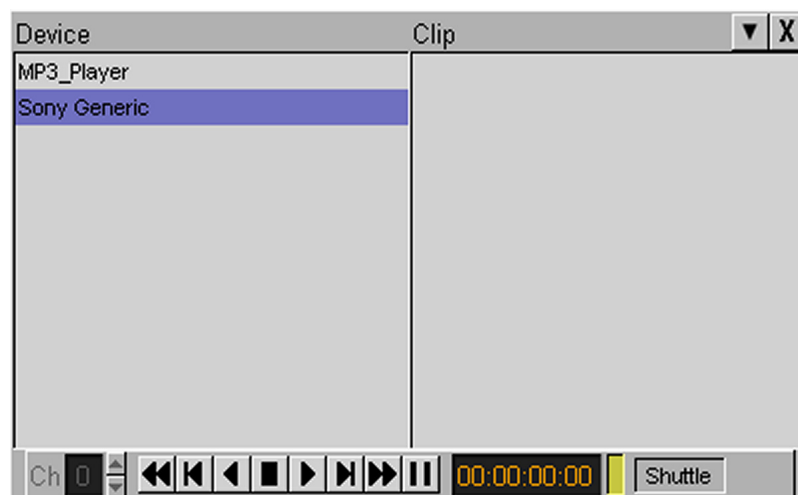
The device manager is located in the config section. Click the Config button in the main menu and select "Device Manager" in the section list. The manager has two lists, the first contain the plug-ins which are currently active, the second lists up all the plug-ins which are installed on your system.



Enable and set up a plug-in

To enable a plug-in, select one of the inactive places in the list of enabled plug-ins, or select the place of another plug-in if you want to replace it. Select thereafter the plug-in you want to enable from the list of installed plug-ins. The entry in the “active plug-ins list” has now changed to the plug-in you selected. To the right, the parameter section of the plug-in has opened. Set the parameters needed and click the save button. Often the program must be restarted for the new settings to take effect. For details on the parameters for each plug-in, see the section describing each of the plug-ins below.

Using the Device Control Unit



The device control unit opens when you click on the button labelled “Device control” located in the lower left corner of the program main window. The Device controller shows to the left a list of the plug-ins which are currently enabled. To the right you will see a list of clips for the plug-in you selected on the left side, if the plug-in have clip capability.

Add a device to the scene stage

To add a device element into your scene, drag the device from the Device controller onto a stage director. An entry with the name of the device is created in the director tree with one or more channels below, some plug-ins have multiple channels. By adding a keyframe on the channel, the device becomes an element in the stage animation process. A special editor for device keyframes opens in to the left of the director tree. It has the following parameters:

- **Name:** You can give the keyframe a name. This must be done if you want to control the keyframe with external controls.
- **Time,** sets the time on the stage time line
- **Clip:** On those plug-ins that have clip capability, you drag the clip from the list in the device controller and onto this field.
- **Begin time,** sets the begin time for the keyframe
- **End time,** sets the end time for the keyframe

Clip Station post

The Viz|Artist clip station post plug-in allows you to post render complex scenes that don't run in real-time onto the clip station DDR (NT-device). It is also possible to add clips from the clip station in the scene (up to 50 channels) and thereafter post render the whole scene with the video clips back to the clip station or to disk. This feature provides a way to build very complex scenes.

Clip Station Control

The clip station control plug-in is a real time playout plug-in that allows you to trigger and play video clips from the Viz|Artist clip station. This can be performed in two ways.

1. The clip station delivers its output on the video input of Viz|Artist (loop-back). The video source is then added to the scene and triggered to start at the wanted time.
2. Viz|Artist triggers the clip station playback and the output is delivered directly to a external keyer together with the graphics from the scene.

ASC VR300

ASC VR300 is a broadcast video server. Video servers are digital audio/video record/playback systems designed for professional broadcast studio use and characterized by a small number of simultaneous channels at very high (240Mbit/s) data rates. These servers are designed for applications such as commercial inserts into television broadcasts where previously automatic cartridge video tape based systems have been used, as well as for action and slow-action replay of recorded events. The extreme data rate requirement is achieved through a combination of RAID (Redundant Array of Inexpensive Disks) storage and fibre-optic disk channel technology (Fibre Channel).

Accom Axess

Clip station interface

Tektronix profile

Video server interface

Sony Generic

The Sony Generic is a video player that Viz can trigger with timecodes. To trigger a playback, drag it from the device manager onto the stage and add a keyframe with the wanted start and stop times. You can also post render a scene to a Sony generic.

The Post Rendering Screen

Post rendering general description

The Post Rendering Screen opens when you click on the Post button in the main menu. All post rendering processes take place here. When it opens, the

list of device plug-ins in the Device Controller is limited to the plug-ins which have a post rendering capability. By using drag and drop, you can load a plug-in onto the post rendering screen to use it for the post rendering process. By default the scene currently loaded will be rendered, but you can change to another scene easily by dragging a scene-icon from the scene database and onto the Post rendering screen. To post render a scene, load the scene and the wanted device-plug-in, set parameters and click the Record-button.

Post rendering screen parameters

- **Scene:** You drag a scene from the scene database and drop it onto the dropzone here to load it for post rendering.
- **Device:** Drag the wanted device plug-in from the list in the Device Controller and drop it onto the dropzone here to post render to it.
- **Clip name:** If the device is of the kind that receives a sequence of clip-images, you enter the name you want for the sequence of clips here.
- **Time Code:** On devices where this is relevant you enter the time code for output to the device here.
- **Begin,** sets the starting frame for the post rendering process.
- **End,** sets the ending frame for the post rendering process.
- **Continue** - If you are post rendering to a single frames format, you can by enabling this option and setting a frame number, continue an interrupted postrendering process from that frame. It will not work if you are rendering clips.⁴

Post Render options

At the upper right side of the post rendering screen you can enable antialiasing and select the degree of multisampling. This makes the rendering take significantly more time especially if you set a high value of multisampling.

Render to disk **Render to Disk device, general description**

The render to disk device allows you to post render a scene to your hard disk, using a format of your choice. It supports the following formats:

- **Raw data,** is files with raw content from the OpenGL frame buffer.
- **SGI Image,** is Silicon Graphics own proprietary format.
- **YUV:** Television broadcast signals and video signals are in YUV space, where Y is luminance and U+V are color differentials.
- **TIFF,** means Tag(ged) Image File Format. It is one of the most popular and flexible of the current public domain raster file formats
- **PNG,** Portable Network Graphics (PNG) format. Was designed to replace the older and simpler GIF format and, to some extent, the much more complex TIFF format
- **PMM,** Portable Pixmap Format is a lowest common denominator image file format.
- **AVI,** The AVI (Audio Video Interleave) format was developed by Microsoft. It is a common format on PC's, and comes with the Windows operating system as part of the "Video for Windows (VfW)" distribution.

To render a scene to disk, you must first set up the Render to Disk device and select the format and settings you want. Thereafter you must open the post rendering screen, load the scene and the render to disk device and start the post rendering process.

Render to disk parameters

To set up the Render to Disk device, open the Device Manager in the Config section. Set the parameters and save the configuration. To apply the setting, the program needs to be restarted.

- **Name,** allows you to set an alternative name for the plug-in, typically a name including the current format settings.
- **Format,** allows you to select the output format to be written to disk.

- **Frame format,**
 - **Full Frame,** will store 50 or 60 full height frames for each second of video. The number depends on the video settings, PAL/50 NTSC/60 frames, and also the size changes with the video format (720x576 for PAL digital and 720x486 for NTSC digital). Normally TV is 25 or 30 interleaved frames, but in this case the post rendering produces twice the amount of pixels needed. An editing system (or wherever you put the images you render) can throw away 50% of the pixels and still produce a "normal" video stream.
 - **Interlaced top/bottom,** will store 25 interlaced frames for each second of video. Each frame consists of 2 fields (half size of PAL or NTSC e.g. 720x288). These two fields are combined line by line. Interlaced top means that the first line of the first field is the first line in the interlaced frame. In Interlaced bottom the first line of the second field of the first line in the interlaced frame. This determines "field dominance".
 - **Fields,** will store 50 fields per second of video. It is the same amount of data as Interlaced top/bottom, but instead of shuffling the data the post rendering process write different files for each field.
- **Components,** sets the color settings of the output images/files.
 - **RGB (24 bit),** creates RGB images. Here a pixel is represented by 3 bytes containing a red, blue and green sample
 - **RGBA (32 bit),** This format is an extension of BI_BITFIELDS where a fourth bitmask is used to define bits in the pixel which correspond to an alpha channel. When displayed on top of other images, RGBA pixels are blended with the background pixel according to the value of this alpha component
 - **RGB+AAA (24bit),** for each image rendered it creates a RGB and an Alpha image.
- **Resize output,** enable this option to resize the output on the X- and Y axis. The device will then render each scene frame in full size and then scale the each image afterwards. If you scale the output down, you will get a "free" antialiasing.
- **Resize X,** sets the resized X-size
- **Resize Y,** sets the resized Y-size
- **Destination Dir,** allows you to select the output directory.
- **Configure AVI compressor,** click this button to configure the AVI compressor if you have selected AVI as your output. For details see next section ["Setting up a video codec" on page 171.](#)

Setting up a video codec

A codec is software that can compress and uncompress audio or video data. When you click on the "Configure AVI compressor" button the codec setup dialog box will be shown. The codec dialog box lists all codecs that are installed on the system. Select the codec you want to use, set its parameters and press enter to save configuration. However, many codecs have specific limitations and will only work on certain types of input. It's not possible to list up all things to be aware of, but here are some:

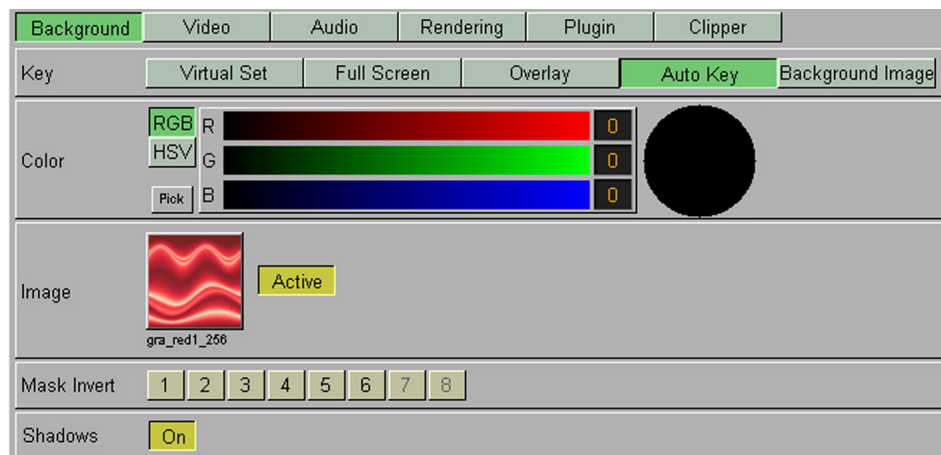
- 24bit uncompressed video is valid but rare. Some programs may have difficulty in reading such files. Try creating an uncompressed video using "RGBA" to create a more common 32bit uncompressed video.
- Not all codecs support both 24bit and 32bit video. You may need to change between "RGB" and "RGBA" to get a specific codec to work.
- Some codecs will work with "RGBA" but will nevertheless store the video as 24bit.
- If you plan to use an "RGBA" video in a program that can handle alpha, you need to make sure that you have KEY- functions on objects you want to have visible. Otherwise the alpha channel will be completely blank, thus making the whole video totally transparent.

- Some codecs requires a handle to the top level application window. The Viz plug-in interface is unable to supply such a handle. As a result the configuration windows of some of the codecs will fail to appear. Also, some codecs will crash unless they get a handle.
- Files created in raw uncompressed mode looks weird in Windows Media Player. This is some oddity of the Media Player. The AVI video file contains pure undithered RGB data, but the Media Player dithers it (often using only 16 colors) for some unknown reason. As I said, the video file is correct, and you should be able to import it into other video editing applications without any problems.

Using the scene setup menu

The setup menu gives you access to a set off parameters relevant to the whole scene setup.

Scene setup Background page



- **Key:** This section allows you to select how the key signal is to be executed.
 - **Virtual set,** is relevant for those who have purchased the virtual set option for Viz|Artist. This mode switches from "Overlay" to "Full screen" when the control mode of the current camera switches from "remote" to "editor". On a virtual set, if you for example want to perform a camera flight with the virtual camera and the real camera is of the stationary kind, the graphics must be switched to full screen to prevent a asynchronous situation where the virtual and the real camera have different position values. As you switch the camera from remote to editor to perform the flight the keyed graphics then automatically switches from "overlay" to "full screen".
 - **Full screen,** sets the whole scene as a key signal. When you create a virtual set scene in Viz newer then PL10 you should make sure that this option is selected. If don't, the virtual blue screen (AKA co_cyc or trash matte) will not be drawn. If you need to load a scene which was created in earlier version than 2.5 PL10 in to a virtual set, make sure you change this parameter and save the scene.
 - **Overlay,** sets as key signal only those containers that have a key function attached.
 - **Auto Key** - (New from version 2.8 pl.1). With this option set, the whole scene will be rendered with key. The key will be rendered correct without having to tweak the signal for some scenes as with the old key technique. This feature uses shader technology and will only work on graphics boards that support Shaders version 3.0 and higher. This key-

ing technique creates correct key for transparent objects and Pre Multiply problems that did occur with the previous keying method will not occur.

- **Background image**, uses the alpha values of the background image as a key signal. Containers with key function attached will also be a part of the key signal in this mode. To use this mode a background image with an alpha layer must be added to the "image" section. See below!
- **Color**, sets the color if the background.
- **Image**, allows you to load an image on the scene background. Drag an image from the image database and onto the dropzone to set it as background.
- **Mask invert**, allows you to set one or more of the mask layers in the scene to be inverted. Normally with no mask inverted, the mask cuts out a hole where it covers the mask target object. With "mask invert" enabled the opposite situation takes place, the mask target is then only visible where the mask covers it.
- **Shadows**, enables or disables the use of shadows in the scene. The shadow function uses two layers from the mask function. With shadows enabled there is only 6 layers available for the mask function. If you have a scene where you need more than 6 mask layers, you can disable shadows. Only in very complex mask configurations are more than 6 layers needed, so normally you can have shadows enabled.

**Scene setup
Video page**

Video menu

	Format		Download Method		
Video 1	RGB	RGBA	Progressive	Interlaced	De-Interlaced
Video 2	RGB	RGBA	Progressive	Interlaced	De-Interlaced
Video 3	RGB	RGBA	Progressive	Interlaced	De-Interlaced
Video 4	RGB	RGBA	Progressive	Interlaced	De-Interlaced

In the video setup menu, you define the format and the download method for the four video sources.

- **Format**
 - **RGB**, is normal RGB
 - **RGBA**, is RGB with an additional Alpha channel.
- **Download Method**, allows you to set the video format for the four video sources.
 - **Interlaced**, In interlaced mode, Viz|Artist is merging two and two fields and downloads the whole interlaced frame to the texture memory. As Viz|Artist is rendering 50 times a second, each frame has to be used two times before it gets replaced by the next frame. When looking at such images in a progressive scan computer monitor, you will see the jaggies (horizontal tearing). However, as long as the video is used exactly fullscreen, this will not be visible on the final video output as the GVO/Digital Voodoo is extracting field information again after the texture is rendered. But as soon as the video is not used fullscreen anymore (rotated, scaled, etc...), it's clear that the original even and odd lines of the input will not map anymore to the corresponding lines in the output. Especially when scaling down the video, some lines will even get lost and some other will be combined together which leads to even bigger and more visible tearing. The big advantage of this

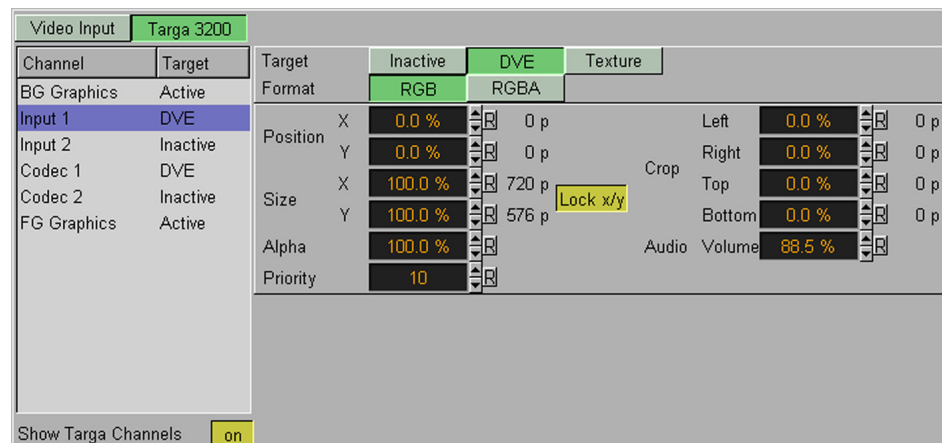
method is of course that the video is keeping its full spatial resolution, i.e. still images will look fine and the result will be a stable image.

- **Progressive**, is another technique Viz|Artist uses for producing the video textures. The method is to use only one field and double the lines in that field: This looks a little better than interlaced, but there is a loss of spatial resolution, i.e. there's lots of vertical blockiness now visible. However, the temporal resolution is kept and especially when the video is scaled down pretty much (like half of its original size where lines are thrown away anyway), the results are much more acceptable than with interlaced mode. The big disadvantage however is that a certain amount of flickering is introduced especially in situations where the video is full screen or even zoomed in.
- **De-Interlaced**, (Only accessible on NT-based systems), uses a combination of interlaced and progressive texturing. Interlaced (field merging) is used for still areas of the picture and line doubling progressive texturing (or even line interpolation) for areas of movement. To accomplish this, motion on a pixel-by-pixel basis must be detected over the entire picture in real-time, e.g. by comparing the luminance value of a pixel with the value two field earlier. As two fields are combined, full vertical resolution is maintained in still areas of the picture, where the eye is most sensitive to detail. In moving areas of the image, the full temporal resolution is kept. However this method is computational very intensive. You will need much processor capacity to run with this download method and still keep the 50 frame rendering rate.

Video menu with Targa video card installed

If the render is configured with a Targa 3200 video card, a extra set of video features is available. Scenes can be designed for the Targa features even if the design station does not have a Targa card installed; the targa video windows will still be represented in the GUI as blue boxes so the designer can place the targa elements.

To access the targa features open the video menu under the Setup item in the main menu. Click the button labeled Targa 3200. The targa editor opens:



There are six input channels in the list. 2 live video, 2 clip, foreground and background graphics. The Targa card can compose 4 layers of video/graphics.

- BG graphics - is the part of the graphics in the scene that have no key function attached. It will always be in the background of the other input channels.
- Input 1 and 2 is live video input.
- Codec 1 and 2 is clip channels that supports DVCPRO25 and DVCPRO50 formats. It is possible to have run separate fill and key clips. The naming

convention is that the alpha/key clip must have the same name with an attached "_key" before the file extension, i.e if the main/fill clip is named "example.dif", the alpha/key clip must have "example_key.dif" as its name. In addition the Targa channel must be set to RGBA.

- FG Graphics is the graphics that will be composed on top of the video sources. All graphics containers with key will be treated as FG graphics.

Possible modes for the channels

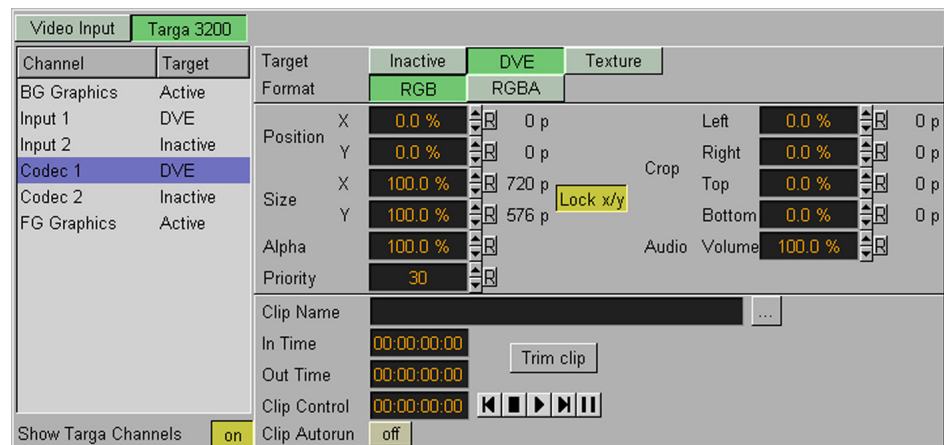
The FG and BG graphics can either be set to mode "Inactive" or "DVE", the rest of the channels can in addition be set to mode "Texture"

- Inactive - The channel will not be shown.
- DVE - The channel will be run on the hardware of the targa video card. The video is composed together with the Viz|Artist graphics. This mode has almost no performance impact on the rendering process.
- Texture - The video is rendered as texture within the Viz-process. In this mode each of the video channels on the targa card is linked to the video sources in the image database. (Input 1 to video 1, input 2 to video 2, Codec 1 to video 3 and codec 2 to video 4. You must use this mode if you want to use the video as a texture on an object. This mode of costs rendering performance.

Parameters for the live video and the clip channels

Both the live video channels and the clip channels have the following parameters, that all can be animated the normal way:

- Position X/Y/Z
- Size X/Y
- Alpha
- Priority - This value decides which elements that are to be shown on top of others. The one with the highest priority is put on top.
- Crop Left/Right/Top/Bottom
- Audio Volume - Sets the volume of the audio output.

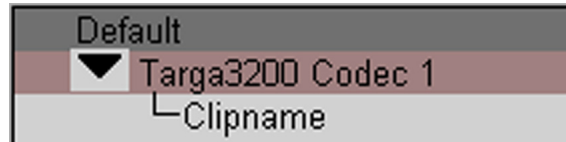


In addition, the Codec channels have parameters relevant to clips:

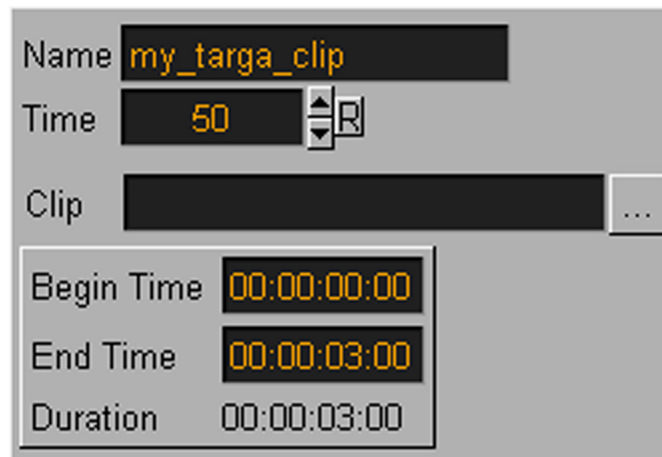
- Format - Decides if the clip is to be treated as RGB or RGBA
- Clip name - browse for the clip you want to play.
- In time - Set the time code for in time
- Out time - Set the time code for out time
- Clip control - allows you to play, stop, pause and go to end/start of the clip
- Trim clip - allows you to trim the clip file in accordance with the time codes given.
- Clip Autorun - When enabled, the clips start to play when the scene is loaded.

Creating Targa clip animation keyframes

The playback of Targa clips can be controlled in the stage with keyframes. A Targa clip keyframe is created by selecting the clip channel in the list and pressing the update button once. An animation channel is then created in the stage.



Select the channel and create a keyframe. The keyframe editor opens to the left.



Here you can browse for the clip file and set begin and end times.

Post render to Targa playable clips

It is possible to render Viz|Artist scenes to Targa clips. This is done in the usual post rendering screen. The following configuration must be done prior to the post rendering:

- The PinnacleDV25 or DVCPRO codec must be installed
- A post rendering device that uses this codec must be configured.

After this is done, normal post rendering can be performed and the clips are directly playable by one of the two clip channels on the Targa card. This can typically be used to save performance in scenes by post rendering background elements to a clipfile and thereafter playing the clipfile instead of the original background.

Separate Fill and Key clip

This post rendering also works when rendering separate fill and key clips, i.e. you can set up the Render to Disk device to create two separate clips, one for fill and one for key.

Note!

Post rendering only works for rendering plain graphics. You can not postrender video input feeds or scenes which include clips that is composed on the targa card. However it is possible to post render clips by playing the clips with the SoftClip plug-in which is capable of playing back clips on a frame-by-frame basis.

Configuration parameters for the targa functionality

There is a set of configuration parameters connected with the targa card. These are found in three different sections:

Targa 3200 section

Parameter	Default	Description
Clip Data Directory	D:\	Default clip directory
Loophrough Delay (EE)	2 frames	Sets the delay between video input and output, i.e. if you are using a video input in DVE mode (not as a texture in Viz), the delay between the video input and output will be exactly(!!) this number of frames
Videoout Ringbuffer	On	Enable/Disable the video out ringbuffer.
Ringbuffer Size	1 frame	Sets the size of the graphics ringbuffer in frames, i.e. changes in the graphics will need up to this number of frames until they are visible on the video output
Audio Delay	2 frames	Sets the audio delay in 0.1 frame steps
Enable embedded audio on Input 1	Off	Enable or disable embedded audio input.
Enable embedded audio on Input 2	Off	Enable or disable embedded audio input.
Frame Accurate Viz Comm.	Off	If this flag is set, all commands sent to Viz will be delayed exactly by ringbuffer size + 1 frames until the result is visible on the video output. So if the commands sent to Viz are genlocked (e.g. using an automation system), frame accurate Viz control is possible. Use this flag with extreme care, as it will also delay the answer sent by Viz, so if your control software requires interaction, do not use this flag. Only use it when all commands are sent with -1 as command number. Note: commands will only be executed on a frame boundary

Scene defaults section

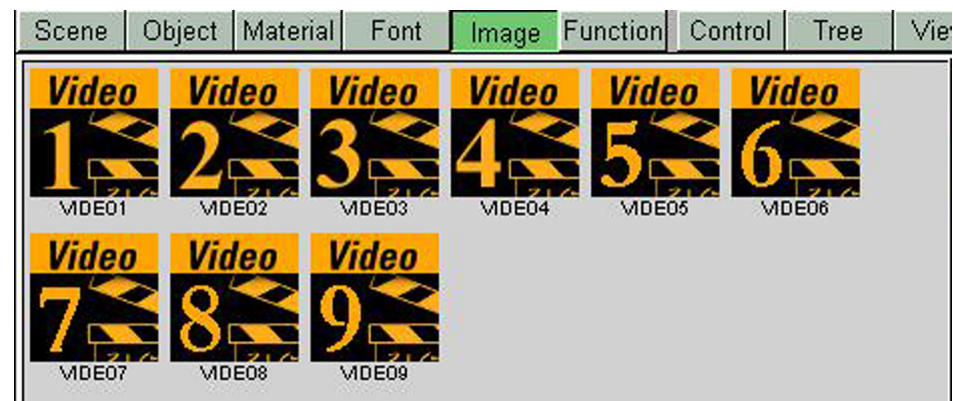
Parameter	Default	Description
Targa 3200 Prior. Channel 1-4	10-40	Sets the initial priority for Targa channels when creating a new scene or loading old scenes without Targa settings
Targa 3200 Target Channel 1-4	Inactive	Sets the initial targets for Targa channels when creating a new scene or loading old scenes without Targa settings. This for example allows you to use internal keying of graphics over a video feed without having to modify/touch each and every old scene. Just set Channel 1 to Targa3200 (should be named DVE and will be changed in the next version) instead of Inactive

Video Output > video output editor

Parameter	Default	Description
Genlock	Freerun	Sets the video reference for the video output ports (Freerun/Blackburst/Digital Input1/Digital Input2). Note that especially when using Video Input you should NOT use Freerun as this can produce artefacts on the video inputs or outputs.
H-Phase	0	Sets the horizontal timing offset compared to the selected reference clock

Extra video menu on CGP hardware from Panasonic

This paragraphs describe the video clip functionality on CGP hardware from Panasonic. On this hardware you will find 9 video sources in Images/BuiltIn.



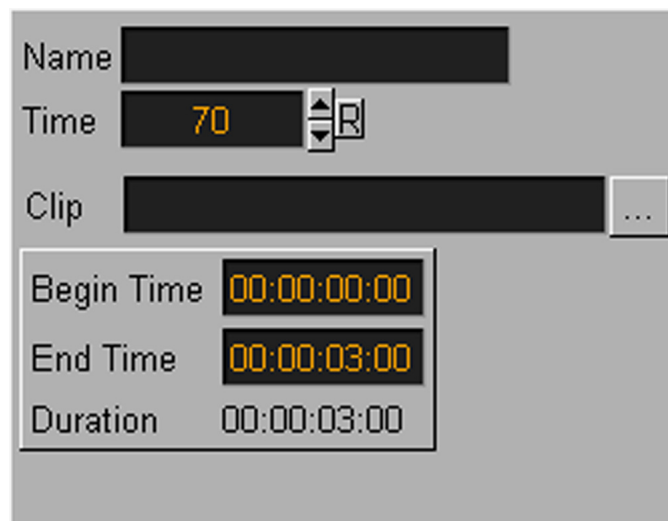
Video source 3 to 9 can be controlled from the CGP video menu.



You can drag any of these video sources into your scene, select the corresponding channel in the video menu, select a clip and click start. The clip will be played back and will be visible on the CGP output, however: you will not see the video on the VGA screen, only a video dummy texture will be shown here.

Add clips to the stage

If you click the update button once when you have selected an active CGP clip channel, an animation channel is created in the stage. To add a clip to the animation channel, select the channel in the stage and click the "Add Keyframe" button. Click the keyframe. To the left an editor opens where you can select a video clip.



Clip file format

The clips files are mpeg4 compressed. They can contain RGB or RGBA. All resolution can be played out. The clip file itself has no suffix, but is identified by a file with the suffix ".toc" which describes for Viz|Artist the content of the file (width, height, length, format etc.). A clip file can only be played back with a matching toc file.

Create clips for the CGP 500

There is available a command line tool which allows to capture the output of the CGP 500 into a clip file.

Performance

The clip playback does not cost any rendering performance.

Limitations

- The decoder chip can decode about 90 Mpixel per second. The video will start to jitter once you want to play back more than this. 90 Mpixels is a bit less than 1.5 full HD size (1080i). So one full 1080i clip and a quarter size 1080i clip can be played back without problems.
- The hard drives where the clips are stored can also be a bottle neck. If the clips can not be read fast enough from the disk, the video will also start to jitter. This does usually not happen, except other processes access the hard drive while you playback your clips. Dedicated drives should be used to store and playback the clips.

Audio From version 2.7 patch level 4 Viz|Artist support playout of audio files. The audio files can be put into the stage and controlled with keyframes and stop points like the normal animation channels.

Installation, configuration and setup

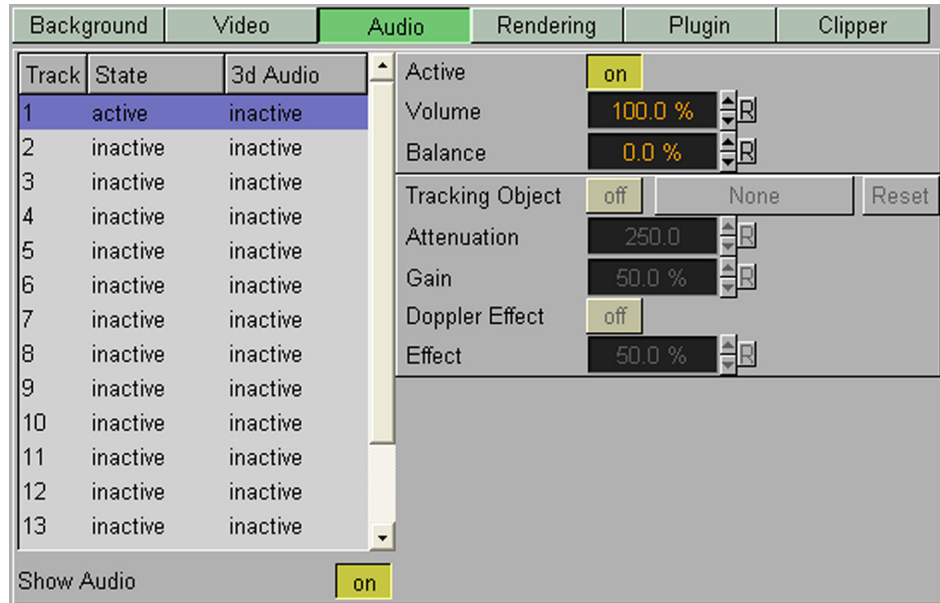
- The audio needs at least directX 8.0 installed on the system.
- In "config/audio" you can switch off the audio for Viz, select the default path for audio files and select the audio device. These are the audio capable devices installed in your system (usually only one). If you have a targa board installed, you can select Targa as audio device and then the audio output will be embedded in the targa video out stream.

Supported audio file formats

At this moment only mp3 and wav file formats are supported. You do not need to think about sampling rates, because Viz|Artist takes care of resampling the audio if needed. It is beneficial, performance wise, if all audio files have the same sampling rate, because then no resampling is needed.

Play back audio files

You only can play back audio via the stage. Go to "scene/setup/audio" and select an audio track.



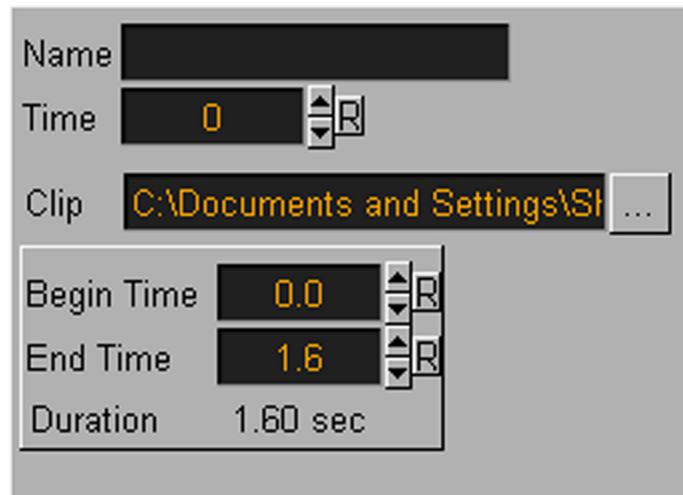
Switch it active, press update and the audio track will show up in the stage.

After adding the track to the stage, select the "clipname" channel and press add keyframe. Now you can select a audio file from the hard disk.



An audio keyframe is shown in the snapshot above.

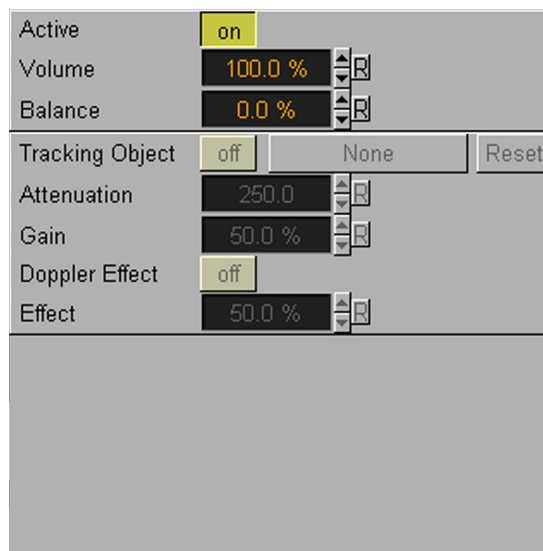
You can set its begin and end time in the audio keyframe editor.



If the end time is higher than the length of the clip Viz will restart the clip and play until the end time is reached.

The audio tracks in the stage can be moved into other directors and react to stop points like any other channel.

Audio parameters



- **Active** - Toggles on/of the audio channel
- **Volume** - Volume for the audio channel. This value can always be changed or animated, also when effects like 3D audio or Doppler are used.
- **Balance** - Sets the audio balance. Can be changed or animated only when the 3D audio effect is off.

3D audio

This effect is activated by dragging a container into the "tracking object" frame of the audio track. The current camera is the "listener" and the container is the object to "listen to". The position and the direction of the object will control the volume and the balance of the track.

- **Attenuation** - controls the "falloff" of the volume depending of the distance of the object to the listener. Attenuation of 100 means that the object has 50% volume when it is 100 Viz units away from the listener. A plane will have a very high attenuation, because it is very loud even when it is far away, whereas a fly would have a very low attenuation.

- **Gain** - The gain parameter can be used to tune the attenuation parameter.

Doppler effect

The doppler effect controls the pitch of the sound by calculating the speed of the object relative to the listener. If the object move towards to camera (listener) the sound will be higher, if it moves away it will be lower. The Effect parameter controls how strong the doppler effect is. 0% means no doppler, 100% will make it very strong. Keep in mind that doppler effect only works when the object or the camera is moving and the speed of these two influence the doppler effect.

Important note

To be able to cue into mp3 files precisely and fast Viz|Artist have to build index files for the mp3 files. These files are created whenever a mp3 files is used the first time. The file has the same name as the audio file, but with the suffix ".idx". This process can take a few seconds, depending on the speed of your system and the size of the file, but it only has to be done once. When the file exists, it will open the index file and use it and this takes hardly no time. So, if you want to change audio files on the fly via external control you must built the index files ahead of time to prevent lockups during OnAir. Make sure that you transfer index files together with the mp3 files when migrating to on-air servers.

Known Issues

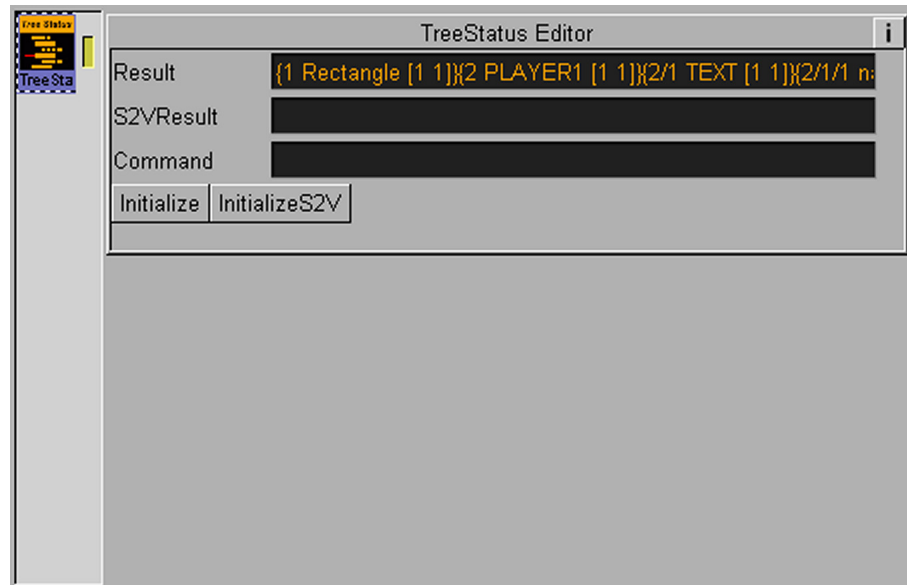
Audio files are not moved together with archives. This will be implemented in the near future. Until then, audio files and index file must be copied separately.

Rendering

Culling	<input checked="" type="checkbox"/> Active	
Screen Area	Safe <input type="text" value="90.0 %"/>	Title <input type="text" value="80.0 %"/>
Auto Key Plane	<input type="text" value="X 0"/> <input type="text" value="Y 0"/> <input type="text" value="Z 0"/>	
	<input type="checkbox"/> Inactive	
Half Height	<input type="checkbox"/> Active <input checked="" type="checkbox"/> Inactive	
Depth Buffer	<input checked="" type="checkbox"/> Active	
Camera	Near <input type="text" value="1.0"/>	Far <input type="text" value="20000.0"/>
Aspect	<input checked="" type="checkbox"/> Configuration <input type="checkbox"/> 4:3 <input type="checkbox"/> 16:9	

- **Culling**, enables/disables a culling function for the scene. The culling function is a performance saving function. With culling enabled, objects whose bounding box are outside of the camera view are not being rendered. This will of course be of great importance to performance in large scenes and virtual sets. Normally the culling function should be enabled.
- **Safe Area**, sets the size of the Safe Area in percent.
- **Title Area**, sets the size of the Title Area in percent.

- **Autokey**, allows you to define a position which serves as a threshold between objects with key and object without key. This position can be delivered by a tracking device carried by the person in the set, or it can be controlled by an operator. The autokey position decides which objects or parts of objects in the scene that are to be parts of the key signal. Only objects positioned between the autokey position and the camera will be rendered with key signal. A typical setup for a virtual set would be that the scene background is put into a separate channel of the external keyer. The objects of the background don't have a key function attached. The objects that are meant to be in front of the persons in the set are placed in another channel/layer of the external keyer and they have a key function attached. If the person in the set moves against the camera the autokey function disables the key function of those objects that now are to be placed behind the person, thereby moving them into the background layer of the external keyer. More detailed information about this function can be found in the documentation for the virtual set option.
- **Half Height**, Viz|Artist will render fields instead of full frames. Normally it renders 50 or 60 full height frames and produce twice the amount of pixels needed. This is done to achieve better quality. If you pick every second line of a full height frame and send it to the output it will look better. Anti-aliasing, texture filtering works "better" if you render full height. So when Viz renders half height, you will lose a little bit of quality, especially on some textures, but it nearly renders twice as fast, because pixel fill is much lower, its half in fact!
- **Depth Buffer**, enables/disables the depth buffer (z-buffer) on the graphics card. Switching it off can improve the performance of your scene on some cards, but it is not sure. Your scene might also look totally wrong without z-buffer, so please consider this as an expert feature. It makes sense sometimes, but on many cases, not.
- **Flickerfilter** - This option works for systems with Targa or Voodoo video boards. The flicker filter reduces flickering on the video output by interpolating between the fields and blurring the images a bit. There are three levels of filtering. Try it out to find the best result for your scene.
- **Camera**, sets the range of the virtual cameras in Viz|Artist
 - Near, sets the close range
 - Far, sets the far range
- **Aspect Ratio**, allows you to override the aspect ratio configuration settings for the current scene. Choose "Configuration" if you want the scene to use whatever is stored in the configuration, or choose 4:3 or 16:9 to override.

plug-in

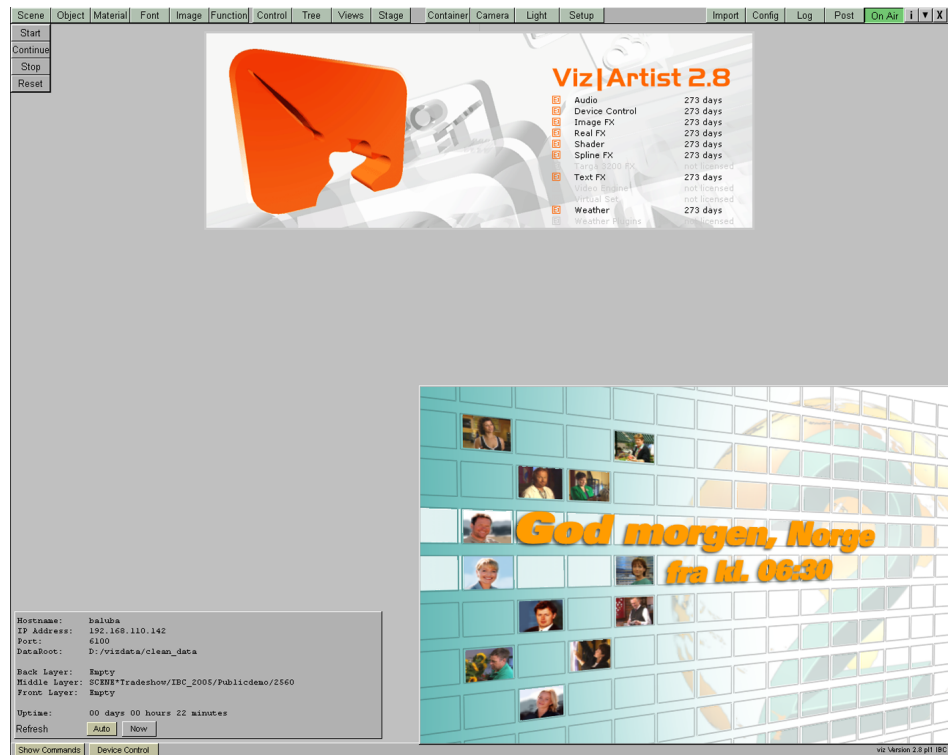
This page is the dropzone for Scene plug-ins. To use a scene plug-in drag it from the scene plug-in section in the Function Database and drop it onto the rectangular dropzone in to the left in this section.

Clipper See: ["Using the Clipper function" on page 69](#)

Virtual Set The virtual set section allows you to set up and save configuration for virtual sets. For documentation see the manual for the virtual set option.

On Air Mode

The On Air button sets Viz|Artist in a control mode where it listens for commands on port 6100 (default, can be changed in Configuration section). In this mode Viz|Artist is basically converted into a Viz|Enigne. The design elements are all hidden.



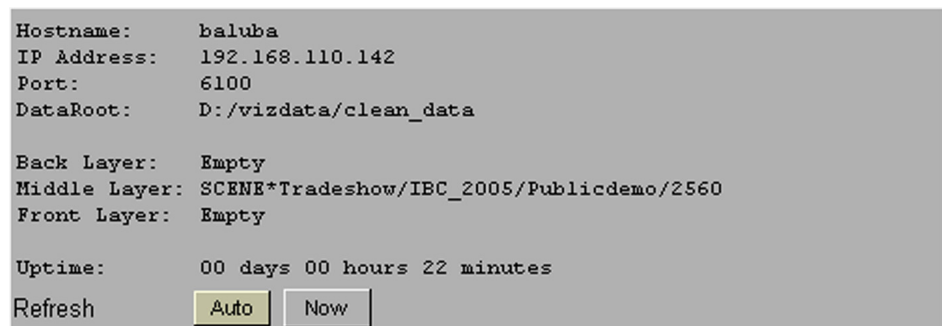
In this mode the system is ready to be controlled by an external application.

In OnAir mode you can perform start, stop, continue and reset for the current scene by using the buttons in the upper left corner:



On Air Info

The On Air info box give you information about the render machine's status when controlled externally.



The info box contains:

- Hostname
- IP Address
- Port - The port that Viz|artist/engine is listening on

- DataRoot
- Back Layer - Shows path to the scene loaded in the Back Layer.
- Middle Layer - Shows path to scene loaded in the Middle Layer.
- Front Layer - Shows path to scene loaded in the Front Layer.
- Uptime - Shows how long the program has been running.
- Auto button - When selected, the info box data will be updated every time a new scene is loaded.
- Now button - If the auto button is not selected, click this button to refresh the information.

External Control Commands

Setup **Configuring the render machine and the control machine**

Viz|Artist have its own control language, which enables external programs to control and execute a wide range of elements. To control Viz|Artist by TCP/IP you must take the following steps:

- Add a service in /etc/services on the SGI: peak-ev 6100/tcp
- Add the same entry: peak-ev 6100/tcp, in the services file on the PC. ('\windows\services' on Windows95/98, and '\Winnt\system32\drivers\etc\services' on WindowsNT/2000).
- Add the SGI's hostname and its IP address to the hosts file on the PC. Example: "174.23.210.5 Viz|Artisthost # Viz|Artist running on".
- Press the 'On Air' button within Viz|Artist or invoke the "Viz|Engine version of the program. Viz|Artist are now in a state where it will receive commands and send answers.

Controller program.

The Viz|Artist external control commands can be used internal in the program (actions) and in scripting in Viz|Content Pilot. In addition the possibility exists to make you own controller program. For that purpose Vizrt have made the DLL, pws98.dll which allows you to create a socket connection to Viz. It gives your program some commands to use for the connection with Viz|Engine, (the examples are in Delphi code):

Make a buffer with commands ans send them:

- function buffer_put (string: PChar):integer;stdcall; external 'pws98.dll';
- function buffer_clear:integer;stdcall; external 'pws98.dll';
- function buffer_send(hostname: pChar; service: pChar): integer;

Request an icon

- function request_icon(hostname:pchar; cmd:pchar; service:pchar; filename: pchar; timeout:integer):integer; stdcall; external 'pws98.dll';

To send multiple commands

- function multiple_cmd_send(hostname:pChar; command: pChar; length:integer; service: PChar):integer;stdcall; external 'pws98.dll';
- Each command in the total command string must be null terminated and the length parameter must be equal to the length of the command buffer.

Send a single command

- function single_cmd_send(hostname: pChar; command:pChar; buffer: pChar; len:integer; service:pChar; timeout:integer):integer;stdcall; external 'pws98.dll';

If the command starts with -1 you will not receive an answer in the buffer. If you start with >=0 you will get an answer in the buffer.

error returns:

- 0 on success
- 2 if internal tcp start-up failed
- 3 if protocol was not found in file protocols
- 4 if service could not be found in file services
- 5 if host could not be found in file hosts
- 6 if TCP port was not found
- 7 if connection to socket was refused
- 8 if file couldn't be sent
- 9 if command couldn't be sent
- 10 error getting answer
- 11 out of memory

Internal Command Syntax

A command consists of 4 parts:

- Number
- Location
- Command
- Arguments.

<#> <location> <command> <argument argument....>

The four parts are separated by one blank. The arguments themselves are also separated by a blank.

Example: 5 SCENE*<scene_name> TREE*<container_name> ADD BOTTOM

This command string can be split up in it's components the following way:

- Number is "5"
- Location is "SCENE*<scene_name> TREE*<container_name>"
- Command is "ADD"
- Argument is "BOTTOM"

Number

Each command must be started with a leading number. This number functions as the identifier of the command string. The number is returned by Viz|Artist in any answer to a command. If you for instance send the command: 99 RENDERER*STAGE START to Viz|Artist, a successful execution of the command will return just the number 99. Any error message will be lead by the number, e.g 99 ERROR <0 RENDERER*STAGE*DIRECTOR*test START> : DIRECTOR not found.

Location

The location element of the total command string, tells Viz|Artist at which level or element of the program, the command is to be executed. If one think of Viz|Artist in terms of a hierarchy, the logic of the location element might be easier to comprehend. The program has some "top levels" like "SCENE" and "RENDERER". RENDERER is the scene currently loaded in the rendering window. If one f.ex wants to start an animation located under one director, one begins the location string with RENDERER, thereafter, STAGE and DIRECTOR+*<directorname>. The command, in this case, "START" is to be executed at that level. It could have been executed at the level above, which is "STAGE", but then the command "START" would have started all directors, since the START-command also apply on that location level. The location string can be described as a kind of "drill down" where you set the scope and relevance of the command element that is to follow. Which commands that are accessible of course vary from location to location.

Location specific syntax rules

- Single location parts are separated by '*'.

- "scene_name", "object_name", "material_name", "font_name" and "image_name" is the full pathname within the actual database. Example: "MATERIAL*artcolors/glow/redglow".
- "container_name" can be:
 - Container number with leading '#'. Syntax: "RENDERER*TREE*#1249". These numbers are created by Viz|Artist and used internally. To obtain the numbers for all containers you can execute the command: "SCENE*<scene_name>*TREE GET". To obtain the number for a single container you can execute the command: "MAIN*LOCATION CONVERT SCENE*<scene_name>*TREE*\$<container_name>". Be aware that the container numbers change each time a scene loads, so the numbers must be parsed out each time you load a scene. When you use these numbers to execute a command, you can skip the location before the container_name. E.g you can write: "#323*TRANSFORMATION*POSITION SET 0 0 0" instead of "RENDERER*TREE*#323*TRANSFORMATION*POSITION SET 0 0 0"
 - Name of container with leading '\$'. Syntax: "RENDERER*TREE*\$container_name". With this method the program searches for the container. This takes more time than the other methods.
 - Name of external control channel with leading '@'. Syntax: "RENDERER*TREE*@<name_controll_channel>". Use of control channels secures a quick access and it make sure the link to the container stays intact, even if the container is moved in the scene tree.
 - Number of tree hierarchy. Syntax: "RENDERER*TREE*1/2/2". To obtain the container numbers, you can run the command, "SCENE*<scene_name>*TREE GET". It parses out all containers in the tree with their hierarchical numbers as well as their name and internal ID-numbers. Subcontainers will get a number format like 2/2, which means that it is the second subcontainer and that its parent is the second container at the tree root.

Commands

The commands vary from location to location. A complete command reference list is in the process of being made. For an example list of frequently used commands, see ["Overview of frequently used commands" on page 189](#)

Command arguments datatypes

Datatype	Explanation
UNKNOWN	invalid type
BOOLEAN	This type is a switch which tells whether something is to be enabled or disabled. Valid settings for this type are: 1/0
ENUM	The enum type accepts certain strings. The valid settings are shown in the argument-description of the command. This STRING's constants are mapped to numeric values during command execution within Viz Artist. There is a special implementation of ENUM-constants, where constants are to be combined with numeric values. These implementations are shown with a "%d" token. Viz Artist expects you to substitute this "%d" with a numeric value.

Datatype	Explanation
UINT	This data-type accepts unsigned numeric values. If the command-interface set min/max ranges, only numbers \geq min and \leq max are allowed! You can send numbers as decimal or as hex (hex digits must be prefixed with "0x" e.g. 0x3F)!
INT	This data-type accepts signed numeric values. If the command-interface set min/max ranges, only numbers \geq min and \leq max are allowed! You can send numbers as decimal or as hex (hex digits must be prefixed with "0x" e.g. 0x3F)!
FLOAT	This data-type accepts numeric values that may include decimal-digits (decimal-separator "."). If the command-interface set min/max ranges, only numbers \geq min and \leq max are allowed!
TIMECODE	A timecode value.
STRING	This data-type accepts any text. Optionally you can enclose the text in double-quotation-marks.
FILENAME	This data-type accepts a valid filename. You should place the filename in double-quotes, if its name includes spaces!
ARGUMENT_LIST	This data-type accepts any argument, no implicit type checking is done.
LOCATION	This data-type accepts any valid Viz-object location.
PVOID	This type accepts a hex-value.
OBJECT_ID	This data-type accepts a valid Viz object id. You need to prefix this object-id with a "#" sign!
RGB	This data-type accepts a color value consisting of the 3 components red-green-blue. Valid values per component are 0 (0x00) to 255 (0xFF).
RGBA	This data-type accepts a color value consisting of the 4 components red-green-blue-alpha. Valid values per component are 0 (0x00) to 255 (0xFF).

Overview of frequently used commands

About the examples

Below follows a set of examples which are intended to be a guide to performing different commands that are commonly used in command scripting. The list is not a complete list of Viz|Artist commands. To simplify the example's syntax, the #-element is omitted.

Use of RENDERER and SCENE as main location.

When a scene is loaded you can use the main location RENDERER instead of SCENE*`<scene_name>`. RENDERER refers to the scene that is currently loaded in the rendering window. When the scene path is long, this can save you some time. In the following examples RENDERER will be used. When you write commands in action keyframes, RENDERER will not work, but you can

use "THIS_SCENE" instead. SCENE*<scene_name> will work in action keyframes.

Open and close scenes

- **Load a scene without showing it in the rendering window:** SCENE*<scene_name> LOAD. The scene with all its objects are then initialized and ready to be set active in the rendering window. This type of preloading is typically used if one is to run a series of scenes. All the scenes are then loaded in advance and set active when scheduled to run.
- **Open a scene and load it directly into the rendering window:** RENDERER SET_OBJECT SCENE*<scene_name>.
- **Close any scene:** Enter the RENDERER SET_OBJECT command with no argument. The scene is closed, but it is still loaded in the memory buffer.
- **Clear all loaded, unused scenes from the memory buffer:** "SCENE CLEANUP"

Other Scene related commands

- **Get all the Viz scenes tree:** SCENE GET_ALL_GROUPS
- **Get the loaded scene in the renderer:** RENDERER GET_OBJECT
- **Set Scene in renderer, load if necessary:** RENDERER SET_OBJECT SCENE*<scene_name>
- **Load Scene to the memory:** SCENE*<scene_name> LOAD

Hierarchy related commands

- **Get Scene Tree:** RENDERER*TREE GET
- **Get the Control Channels:** RENDERER*TREE*CONTROL_CHANNEL GET
- **Get Visibility Status:** RENDERER*TREE*<container_name>*ACTIVE GET
- **Change Visibility Status:** RENDERER*TREE*<container_name>*ACTIVE SET <OnOff>
- **Get Key Status:** RENDERER*TREE*<container_name>*KEY GET
- **Change Key Status:** RENDERER*TREE*<container_name>*KEY*DRAW_KEY SET <OnOff>
- **Get Text string:** RENDERER*TREE*<container_name>*GEOM*TEXT GET
- **Get Name of Image:** RENDERER*TREE*<container_name>*TEXTURE*DATA GET

Access and start animations

- **Start all animations at the stage:** "RENDERER*STAGE START"
- **Start all animations at the stage in reverse:** "RENDERER*STAGE START REVERSE"
- **Start a specific director:** "RENDERER*STAGE*DIRECTOR*<director_name> START"
- **Stop all animations at the stage:** "RENDERER*STAGE STOP"
- **Stop a specific director:** "RENDERER*STAGE*DIRECTOR*<director_name> STOP"
- **Continue all animations at the stage:** "RENDERER*STAGE CONTINUE"
- **Continue a specific director:** "RENDERER*STAGE*DIRECTOR*<director_name> START"

Directors, Keyframes and timelines

- **Add a director:** "RENDERER*STAGE ADD <director_name>"
- **Move an animated object to a another director:** "RENDERER*STAGE*DIRECTOR*<director_name>*OBJECT SET SCENE*<scene_name>*TREE*<container_name>"
- **Change a x-value on a position keyframe:** "RENDERER*TREE*<container_name>*TRANSFORMATION*POSITION*ANIMATION*KEY* $\$$ <key_name>*VALUE*X SET <float>".

- **Change all values on a position keyframe:** "RENDERER*TREE*<container_name>*TRANSFORMATION*POSITION*ANIMATION*KEY*\${key_name}*VALUE SET <float> <float> <float>".
- **Change all diffuse values on a material keyframe:** "RENDERER*TREE*<container_name>*MATERIAL*DIFFUSE*ANIMATION*KEY*\${key_name} SET <float> <float> <float> (Between 0 and 1)
- **Set the time for a keyframe:** "RENDERER*TREE*<container_name>*ANIMATION*KEY*\${key_name}*TIME SET <time>".
- **Set the animation offset for a director:** "RENDERER*STAGE*DIRECTOR*<director_name>*OFFSET SET <time>".
- **Set the animation offset for an object in the stage:** "RENDERER*TREE*<container_name>*ANIMATION*KEY*\${key_name}*TIME SET <time>".

Actions

- **Add an action:** "RENDERER*STAGE*DIRECTOR*<director_name>*ACTION ADD <action_name> (string)"
- **Add a keyframe to an action:** RENDERER*STAGE*DIRECTOR*<director_name>*ACTION*<action_name> ADD". From Viz|Artist you get in return a #-number. Use this when you give the keyframe a name.
- **Set a name on an action keyframe:** "RENDERER*STAGE*DIRECTOR*<director_name>*ACTION*<action_name>*KEY*\${key_name}*NAME SET <string>"
- **Give a action keyfram a value:** "RENDERER*STAGE*DIRECTOR*<director_name>*ACTION*<action_name>*KEY*\${key_name}*VALUE SET <string>"
- **Set the time for an action keyframe:** "RENDERER*STAGE*DIRECTOR*<director_name>*ACTION*<action_name>*KEY*\${key_name}*TIME SET <time>"
- **Set the mode for an action keyframe Command or Task:** RENDERER*STAGE*DIRECTOR*<director_name>*ACTION*<action_name>*KEY*\${key_name}*MODE SET COMMAND". (or TASK)

Jump the animation timeline

- **Go to a point on the stage main timeline:** "RENDERER*STAGE SHOW <time>"
- **Go to a point on a director timeline:** "RENDERER*STAGE*DIRECTOR*<director_name> SHOW <time>"
- **Go to a stop or tag on the timeline:** RENDERER*STAGE SHOW <tag_name/stop_name>

Create an animation channel

Creating an animation channel can be done mainly in two ways:

Method 1

To create an animation channel remotely you have to create a minimum of two keyframes. Do the following:

- **Set the timeline at the point where you want the first keyframe to be created:** "RENDERER*STAGE SHOW <time>"
- **Set the values for the object. For instance:** "RENDERER*TREE*<container:name>*TRANSFORMATION*POSITION SET <float> <float> float"
- **Create the key at the same time as you set the timeline to:** "RENDERER*TREE*<container_name>*ANIMATION*KEY UPDATE <time>"
- **Set the timeline at the point where you want the second keyframe to be created:** "RENDERER*STAGE SHOW <time>"

- **Set the values for the object. F.ex:** "RENDERER*TREE*<container:name>*TRANSFORMATION*POSITION SET <float> <float> float"
- **Create the key at the same time as you set the timeline to:** "RENDERER*TREE*<container_name>*ANIMATION*KEY UPDATE <time>"

Method 2

- **Create the keyframes:** RENDERER*TREE*<container_name>*TRANSFORMATION*POSITION*ANIMATION*POSITION ADD <time>"
- The program returns a #-id for each keyframe you create. Use them to give the keyframes names.
- **Name the keyframes:** "RENDERER*TREE*<container_name>*ANIMATION*KEY*#-ID*NAME SET <string>"
 - **Update the newly created keyframes:** RENDERER*TREE*<container_name>*ANIMATION*KEY UPDATE"
 - **Give the keyframes values:** "RENDERER*TREE*<container_name>*TRANSFORMATION*POSITION*ANIMATION*POSITION*KEY*\$<key_name>*VALUE SET <float> <float> <float>"
 - **Change the time for the keyframes if necessary:** "RENDERER*TREE*<container_name>*TRANSFORMATION*POSITION*ANIMATION*POSITION*KEY*\$<key_name>*TIME SET <time>"

Set container transformation values

- **Set the X-value of a container:** "RENDERER*TREE*<\$con1*TRANSFORMATION*POSITION*X SET <float>"
- **Set the X, Y and Z value on a container:** "RENDERER*TREE*<container_name>*TRANSFORMATION*POSITION SET <float> <float> <float>"
- **Set a relative rotation on a container X-axis:** "RENDERER*TREE*<container_name>*TRANSFORMATION*ROTATION*X SET_RELATIVE <float>"
- **Set an absolute rotation on a container X-axis:** "RENDERER*TREE*<container_name>*TRANSFORMATION*ROTATION*X SET <float>"
- **Set an absolute rotation on all values:** "RENDERER*TREE*<container_name>*TRANSFORMATION*ROTATION SET <float> <float> <float>"
- **Scale a container on the Y-axis:** "RENDERER*TREE*<container_name>*TRANSFORMATION*SCALING*Y SET <float>"
- **Scale a container on all axes:** "RENDERER*TREE*<container_name>*TRANSFORMATION*SCALING SET <float> <float> <float>"
- **Enable billboard or follow path for a container:** "RENDERER*TREE*<container_name>*TRANSFORMATION*AUTO_ROTATION SET <enum>". Valid range: BILLBOARD, FOLLOW_PATH, NONE.
- **Set the axes for the billboard mode:** "RENDERER*TREE*<container_name>*TRANSFORMATION*BILLBOARD_AXIS SET <enum>". Valid range: X, XY or Y.

Set/change an image/texture in a container

- **Set/change an image/texture on a container:** "RENDERER*TREE*<container_name>*IMAGE SET IMAGE*<image_name>"
- **Remove an image/texture from a container:** "RENDERER*TREE*<container_name>*TEXTURE DELETE"
- **Scale a texture:** RENDERER*TREE*<container_name>*TEXTURE*SCALING SET <float> <float>"

Set/change a material in a container

- **Load a material from the material database:** REN-
DERER*TREE*<container_name>*MATERIAL SET MATE-
RIAL*<material_name>
- **Set/change material by giving the RGB values:** "REN-
DERER*TREE*<container_name>*MATERIAL*EMISSION SET <float>
<float> <float>". This example changed the emission values. Substitute
"EMISSION" with "AMBIENT", "DIFFUSE" or "SPECULAR" to set the other
material properties.
- **Set the shininess of a containers existing material:** REN-
DERER*TREE*<container_name>*MATERIAL*SHININESS SET <float>
- **To set the alpha value of a containers existing material:** REN-
DERER*TREE*<container_name>*MATERIAL*ALPHA SET <float>.
- **Set the unlit base color of an object:** "REN-
DERER*TREE*<container_name>*MATERIAL*COLOR SET <float>
<float> <float> <float>" (RGBA). This is the color with no shading and
which is visible with no light sources enabled. It only shows when lighten-
ing is disabled.
- **Enable/disable lightening for a color:** "REN-
DERER*TREE*<container_name>*MATERIAL*ENLIGHTENED SET <bool-
ean> (1 or 0)"

Functions

- **Add the alpha function to a container:** "REN-
DERER*TREE*<container_name>*ALPHA SET <float>".
- **Get value of existing alpha function:** "REN-
DERER*TREE*<container_name>*ALPHA*ALPHA GET".
- **Set value of existing alpha function:** "REN-
DERER*TREE*<container_name>*ALPHA*ALPHA SET <float>".
- **Add an extrusion function to a container:** "REN-
DERER*TREE*<container_name>*EXTRUSION SET".
- **Delete an extrusion function from a container:** "REN-
DERER*TREE*<container_name>*EXTRUSION DELETE".
- **Set the extrusion depth:** "REN-
DERER*TREE*<container_name>*EXTRUSION*PATH*DEPTH SET
<float>"
- **Set the expert function on a container:** "REN-
DERER*TREE*<container_name>*EXPERT SET EXPERT*Expert"
- **Set the rendering mode in expert function to multiply:** "REN-
DERER*TREE*<container_name>*EXPERT*RENDER_MODE SET MULTI-
PLY".
- **Enable the show backside option in expert function:** "REN-
DERER*TREE*<container_name>*EXPERT*BACKSIDE SET 1"
- **Add a key function to a container:** RENDERER*TREE*<container<-
name>*KEY SET KEY*key"

Fonts and text

- **Set a font:** "RENDERER*TREE*<container_name>*FONT SET
<font_name>"
- **Set a text on a container holding a font:** REN-
DERER*TREE*<container_name>*GEOM*TEXT SET <string>
- **Set a font kerning:** "REN-
DERER*TREE*<container_name>*GEOM*KERNING SET <float>

Create containers and load objects

- **Add a new container below an existing:** "REN-
DERER*TREE<container_name> ADD BOTTOM"
- **Add a new container above an existing:** "REN-
DERER*TREE<container_name> ADD TOP".

- **Add a new container as subcontainer to an existing:** "RENDERER*TREE<container_name> ADD RIGHT".

If successfully executed Viz returns your command number and in addition the new container's tree-number. Use this number to refer to the container when you give it a name.

- **Give a container a name:** RENDERER*TREE*<container_name>*NAME SET <string>. As <container_name> you here use the tree-number you got when the container was created.
- **Load a "built in" object onto a container:** "RENDERER*TREE*<container_name>*GEOM SET BUILT_IN*GEOM*<object_name>". To get hold of all the built in objects you can run the following command: "GEOMETRY GET_BUILT_IN"
- **Load a imported geometry onto a container:** RENDERER*TREE*<container_name>*GEOM SET <object_location>" To get list the geometry names, run the command: "GEOMETRY GET/<path>". To list up all folders in the object tree, run the command: "GEOMETRY GET_ALL_GROUPS"

Light and Cameras

- **Enable a lightsource:** "RENDERER*<light_name>*ACTIVE SET <boolean>" (0 or 1). "Light_name" is "LIGHT1" to "LIGHT8".
- **Find out which type of light a light source is:** "RENDERER*LIGHT3*TYPE GET".
- **Set the light type:** "RENDERER*<light_name>*TYPE SET <enum>". Valid arguments: LOCAL, INFINITE OR SPOT.
- **Set a light source position:** "RENDERER*<light_name>*POSITION SET".
- **Set a light source's direction:** "RENDERER*<light_name>*ROTATION SET <float> <float> <float>".
- **Enable a tracking object for light source direction:** "RENDERER*<light_name>*ROTATION*TRACK*OBJECT SET SCENE<scene_name>*TREE*<container_name>".
- **Enable a tracking object for light source position:** "RENDERER*<light_name>*POSITION*TRACK*OBJECT SET SCENE<scene_name>*TREE*<container_name>".
- **Set the color of a light source:** "RENDERER*<light_name>*COLOR SET <rgb>"
- **Set the spotlight attenuation:** "RENDERER*<light_name>*EXPONENT SET <float>".
- **Select a camera view:** "RENDERER SET_CAMERA <int>", range 1-16.
- **Set at camera position:** "RENDERER*<camera_name>*POSITION SET <float> <float> <float>".
- **Enable a tracking object for a camera direction:** "RENDERER*<camera_name>*ROTATION*TRACK*OBJECT SET SCENE*<scene_name>*TREE*<container_name>".
- **Remove a tracking object for camera direction:** "RENDERER*<camera_name>*ROTATION*TRACK*OBJECT SET NONE".
- **Deactivate/activate a camera position tracking object:** "RENDERER*<camera_name>*ROTATION*TRACK*ACTIVE SET <boolean>", (0 or 1).
- **Pan a camera:** "RENDERER*<camera_name> PAN SET <float>"
- **Change between orthogonal and perspective view:** "RENDERER*<camera_name>*VIEW SET <enum>". Valid arguments: PERSPECTIVE, ORTHOGONAL.

Images and snapshots

- **Take a snapshot of render output and save it in the image database:** "IMAGE*<image_name> SNAPSHOT RGB".
- **Export an image from the image database:** "IMAGE*<image_name> EXPORT <path+file_name> JFIF". With no path specified, the image will

be placed in the Viz "home-direcory". You can specify alternative locations relative to that directory. Example: ".././snapshots/<file_name>". Don't give the file an extension, the export feature does that. To export to other formats, substitute "JFIF" with "SGI" or "TIFF".

- **Create a preview image of a scene at defined state:** RENDERER PREVIEW <scene_name> <filename> director1 <timecode> director2 <timecode> directorX <timecode> ". <scene_name> can be CURRENT which is the scene currently in the render, or a normal scene path specification like SCENE*<scene_name>. Timecode can be time in seconds or frames.

Archive scenes

Archiving scenes is done in two steps. First you add the scenes you want to archive to the "archive pool". Second you execute the archiving.

- **Add scene to the archive pool:** "ARCHIVE*SCENE ADD SCENE*<scene_name>".
- **Archive the contents of the "Archive pool":** "ARCHIVE STORE "path+archive_name"", Example: "../data/archive/studio2"

Material database

- **Create a new material:** MATERIAL NEW. On success Viz|Artist returns the # you sent and a #-id for the new material.
- **Save and name the new material in the material database:** "MATERIAL*<the #-number> SAVE_TO_DATABASE <material_name>".
- **Set the diffuse material properties for the new material:** MATERIAL*<material_name>*DIFFUSE SET <float> <float> <float>. Range is between 0 and 1. To set properties for the other elements, substitute "DIFFUSE" with "AMBIENT", "SPECULAR" or "EMMISSION".
- **Set the shininess of a material in the database:** MATERIAL*<material_name>*SHININESS SET <float>. Range 0-100.

Change material on a container

Note: For the following commands the <r>, <g> and values are normalized (that is, each is a decimal between 0-1 including, i.e. 0.73 or 1).

- **Change Material Ambient:** RENDERER*TREE*<container_name>*MATERIAL*AMBIENT SET <r> <g>
- **Change Material Specular:** RENDERER*TREE*<container_name>*MATERIAL*SPECULAR SET <r> <g>
- **Change Material Diffuse:** RENDERER*TREE*<container_name>*MATERIAL*DIFFUSE SET <r> <g>
- **Change Material Emission:** RENDERER*TREE*<container_name>*MATERIAL*EMISSION SET <r> <g>

Bounding box related commands

- **Show or hide the bounding box:** RENDERER SHOW_BOUNDING_BOX <OnOff>
- **And then for a specific node:** RENDERER*EDITED_OBJECT SET SCENE*<SceneName>*TREE*<container_name>
- **The x,y,z coordinates and extents for a node:** RENDERER*TREE*<container_name>*BOUNDING_BOX GET

Appendix

List of shortcuts in Viz|Artist

Camera shortcuts

Position X	P + left mouse button
Position Y	P + middle mouse button
Position Z	P + right mouse button
Position X/Y	P + left and middle mouse button
Position X/Z	P + left and right mouse button
Pan	I + left mouse button
Tilt	I + middle mouse button
Twist	I + right mouse button
Pan/tilt	I + left and middle mouse button
Zoom	U + left mouse button
Orbit X	O + middle mouse button
Orbit Y	O + left mouse button
Close in/out	O + right mouse button
Orient against selected object	Press the T key

Spline object shortcuts

Go to start position on drawing pad	Home key
Set Zoom to 1	CTRL+Home
Switch handles on/off on selected points. Bezier/Linear	Esc
Delete selected point. With no point selected, the point with the highest number	Delete
Delete the whole spline	CTRL+Delete
Toggle bounding box on/off	Space
Set zoom factor from 1-9	CTRL+1-9
Set zoom factor, from 0.1-0.9	CTRL+SHIFT+1-9

Toggle on/off handle display on selected points	A-key
Set/delete anchor point at cursor position	CTRL+A
Toggle color scheme between white or black	B-key
Close/open spline	C-key
Copy points onto clipboard	CTRL+C
Enable/disable spline info	I-key
Toggle handles on/off on all points. Bezier/Linear	L-key
Create another polygon.	Press Insert-key and click the pad
Cut spline at cursor position	S-key
Change to "cutting mode"	SHIFT + S
Select all points on spline	CTRL + S
Paste points from clipboard	CTRL +V
Zoom out	X
Zoom in	Z
Set zoom factor to 1	CTRL + Z
Cut selected points to clipboard	CTRL + X

Mouse/key on all value fields

By dragging the mouse across a value field you change its value.

Hold Alt-Key and drag	Changes the value by a factor of 10
Hold Ctrl-Key and drag	Changes the value by a factor of 0,1
Hold Ctrl- and Shift-Key and drag	Changes the value by a factor of 0,01
Hold Alt- and Shift-key and drag	Changes the value by a factor of 100

Text kerning

Change character-kerning	In the text editor, place the cursor between the character pair you want to change kerning for, press Alt-key and use right arrow-key to increase and left arrow-key to decreases. If you press Alt- and Shift key simultaneously, the change is bigger.
Change word-kerning	Same as for character kerning, but position cursor between two words.

Reset buttons in editors

Double click on reset button	Resets values in the whole group of values, e.g both X- Y- and Z-values
------------------------------	---

Object pools

Middle mouse button pressed and move cursor up/down	Scrolls the object list
Copy element	Ctrl + drag and drop

In stage

Increase scale on stage timeline	Z-key. Hold key to Zoom
Decrease scale on stage timeline	X-key. Hold key to Zoom
Pan the stage window	Middle mouse button and drag horizontally
Pan the timeline	Left mouse button and drag horizontally
Select keyframe	Left click
Select multiple keyframes	Hold Shift-key and select with left mouse button
Select a keyframe and all after to the end of channel	Hold Alt-key and left click

Render window

Show advanced performance information	Click on P-button with middle mouse button
Copy container	Hold Ctrl + drag and drop
Move with locked X or Y axis	Hold shift + drag with left mouse button

Main menu

Open Scene database	F1
---------------------	----

Open Object database	F2
Open Material data- base	F3
Open Font database	F4
Open Image database	F5
Open Function data- base	F6
Open Control channel view	F7
Open Tree view	F8
Open Views	F9
Open Stage	F10
Set Viz Artist in on-air mode	F12

Tree

Copy a container	Hold Ctrl-key + drag and drop
Copy multiple contain- ers	Select multiple containers. Hold Ctrl-key + drag and drop
Create multiple copies	Hold Ctrl-key drag with left mouse button pressed. Keep holding and press right mouse button when cursor is at the wanted position. Each time you right click a new copy will be made
Add a property to mul- tiple containers	Select multiple containers and drag and drop the property onto the selection. This is very useful together with the search function. Search>press "select all">drag and drop the property onto the selection.
Remove a property from multiple contain- ers.	Select multiple containers, drag and drop the property from one of the containers onto the recycle bin.
Open/close all groups inside a group	Hold the Alt-key and click on the open/close arrow. Works like the +/- buttons but for one group only

Index

Symbols

@ 33

A

actions 163
 external control 191
 alpha channel 11, 57, 78
 alpha editor 29
 animation channels 36
 archive 15
 archive button 15
 attenuation 71
 axis interpolation 165

B

billboard 53
 bounding box 62
 built in objects 8

C

camera editor 33, 71
 center shift 72
 clip capability 39
 clip functionality 41
 clock 59
 colors 24
 combine images 47
 config 5
 container button 13
 container editor 26
 container plugins 47
 control channels 33

D

data root directory 4, 16
 deactivating an object reference 54
 delete a container 51
 delete elements from container 52
 device controller 40
 device manager 40
 device plugins 39
 devices 39
 director 36
 drop shadow 58

E

expert editor 30
 expert function 12
 export an image 47
 external control 16

extrusion editor 31
 extrusion function 12, 67

F

face orientation 18, 28
 follow path 53
 font database 5, 9
 font viewer 26
 full screen 172
 function database 5, 10

G

geometry editor 28
 grid value 35
 group function 11

H

hide container 52

I

image database 5, 9
 image editor 25
 image export 10
 import 5, 14
 import problems 49
 imported images 9
 imported objects 8
 input devices 39
 installing plugins 4
 installing program files 3
 inverted mask 67

K

kerning 47, 58
 key 11
 key editor 29
 keyframes 37
 add keyframe 162
 delete keyframe 162

L

level of detail 74
 license 2
 light
 Infinite light 71
 local light 71
 Spot 71
 light editor 32, 70
 light settings 32
 Line spacing 58
 locked 22
 logs 5

M

- main menu 5
- mask editor 30
- mask function 12, 67
- mask invert 173
- material
 - alpha 46
 - ambient color 45
 - base color 46
 - diffuse color 45
 - emissive color 46
 - shininess 46
 - specular color 45
- material database 5, 9
- material editor 24
- merge containers 52
- multiple drag & drop 51

O

- object axis position 53
- object database 5, 8
- object size 54
- object timeline 36
- orthogonal 24, 62, 64, 72
- output to devices 39
- overlay 172

P

- parallel projection 33
- parent-child hierachy 11
- performance 55, 56, 57, 62, 66, 71, 74, 75
- perspective 33
- playback-devices 39
- plugins 14
- post rendering 5, 39
- post rendering screen 41
- primitives 74
- program files 3
- projector 13, 32, 69
- projector editor 32

R

- reactivating a reference 54
- rendering area 24
- renderoptions 66
- rotation
 - absolute 53
 - relative 52

S

- save compound object 52
- scaling 52
- scene database 5, 6
- scene plugins 14, 47

- Scene tree 22
- search 24
- shading 55
- shading angle 18, 28
- shadow 30
- shadow caster 12
- shadow function 67
- shadow receiver 12, 31
- shortcuts 72
- snap to grid 35, 63
- soft shadows 58
- split 26
- split container 52
- stage editor 33

T

- tags 163
- tessellation 74
- text box 60
- text editor 29
- texture
 - clamp 56
 - repeat 56
- texture editor 28
- texture environment
 - blend 56
 - decal 56
 - modulate 56
- texture fonts 57
- texture mapping
 - linear 55
 - reflect 56
 - vertex 55
- texture quality
 - linear 56
 - mipmap 56
 - pixel 56
- ticker 82
- timeline 36
- tracking 33, 73
- transfer properties 54
- transformation editor 27
- tree 22
- two images in one container 57

U

- unify 18
- Upgrading program files 4

V

- vertices number 54
- video stills 9
- views 64
- virtual camea 33

W

Word spacing 58

Z

z-sort 12