

Blender IA/IGS Exporter (Bigex)

User Manual

HenningBR218

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

1 Contents

1 Contents.....	2
2 Table of pictures.....	3
3 Summary.....	4
4 Content and goal of this manual.....	4
5 Copyright, Licence, Warranty, Contact, Credits.....	5
6 Introduction.....	5
7 About Bigex.....	6
7.1 Purpose of usage.....	6
7.2 General.....	6
7.3 Feature overview.....	6
7.4 History.....	7
8 Installation.....	8
8.1 Supported program versions.....	8
8.2 Required programs.....	8
8.3 Installing the Bigex scripts.....	8
8.4 Blender example files.....	9
8.5 Further important files.....	10
9 Usage.....	11
9.1 Exportable objects.....	11
9.2 Starting the graphical user interface (GUI).....	11
9.3 Graphical user interface (GUI).....	13
9.3.1 Input fields and buttons of the Bigex user interface.....	13
9.4 Export process.....	15
10 Creating objects for the IA/IGS export in Blender.....	17
10.1 Object types.....	17
10.2 Object orientation.....	17
10.3 Object hierarchies.....	17
10.4 Known issues at import into Rail Works.....	18
10.5 Edge marker.....	18
10.6 Modifier.....	19
10.7 Animations.....	20
11 UV coordinates and Texture layer.....	21
11.1 Overview.....	21
11.2 UV coordinate Layer.....	23
11.3 Texture layer.....	23
11.4 Backward compatibility with old IGS export plugin.....	26
11.5 Shader.....	27
12 Data modification during export.....	32
12.1 Basic functionality.....	32
12.2 Contents of the Bigex Export Data Modification (EDM) files.....	33
12.3 Modification of object attributes.....	33
12.3.1 Search domain header.....	33
12.3.2 Search and replace expressions.....	34
12.3.3 IGS- and IA-object lists.....	35
12.3.4 Attributes of objects.....	36
12.4 Action expressions.....	37
12.4.1 Merging of mesh objects.....	37
13 Running Bigex in command line mode.....	39
14 Sending problem reports.....	41
15 Annex.....	42
15.1 Mapping of Blender data on the IA/IGS format.....	42
15.1.1 Blender Input fields for the IGS format.....	42
15.1.2 Blender input fields for the IA format.....	47
15.2 References.....	48

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

2 Table of pictures

Picture 1.1: Title Picture 1.....	1
Picture 1.2: Title Picture 2.....	1
Picture 9.1: Exported objects.....	11
Picture 9.2: Starting the user interface 1.....	12
Picture 9.3: Starting the user interface 2.....	12
Picture 9.4: User interface.....	13
Picture 10.1: Object hierarchies.....	18
Picture 10.2: Edge Marker.....	19
Picture 10.3: Modifier.....	19
Picture 10.4: Animation parameters.....	20
Picture 11.1: UV Texture.....	21
Picture 11.2: Texture Layer.....	22
Picture 11.3: UV Layer.....	23
Picture 11.4: Texture Layer Stack.....	24
Picture 11.5: Active Texture Layer.....	24
Picture 11.6: Render Stages.....	25
Picture 11.7: Settings for export with both exporter plugins.....	26

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

3 Summary

The Blender IA/IGS Exporter 2.1 (*Bigex*), is a Python Plugin for the very powerful 3D modelling and animation suite Blender. *Bigex* allows the export into Kuju's own binary intermediate formats IGS (3D model geometry) and IA (model animation). These file formats are required to import 3D models and, as far as existing, their animation into Rail Simulator and Rail Works.

Note: The Blender IA/IGS Exporter version 2.1 does support the export into the both file formats.

By the build-in modification function it is possible to set all data fields of the IA and IGS file to arbitrary values. By this the user has the full control on all data written into the IA and IGS files. This is in particular important for all data fields which can not be set via the Blender user interface.

4 Content and goal of this manual

This is the manual for the Blender IA/IGS Exporter 2.1. Goal of this manual is to supply all necessary information for creating successfully an IA/IGS file with *Bigex*. All handling steps and settings which are required to create simple and also more complex IA/IGS files, will be described in compacted form.

As far as required for the overall understanding, background information will be supplied. Complex circumstances will be explained in more detail.

Precondition for the successful creation of 3D models and animations are at least a basic understanding in all of the following areas:

- **Common:** Shader, basic functionality of shader
- **Blender:** Navigation in Blender menus, creation of simple 3D objects, creation of UV layer (unwrapping), creation and handling of materials in Blender
- **Rail Simulator / Rail Works:** Rail Simulator / Rail Works own shader, exact shader names, shader parameters

All of those topics are assumed as known here in this manual.

Important note: This manual

- is **not** a Blender instruction for beginners
- contains **not** a step-by-step guide for the creation of Rail Simulator / Rail Works models or animations
- is **not** a reference book for Rail Simulator / Rail Works shader names, functionality and effects

But all these informations are available in the internet under the links listed in section *15.2 References*.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

5 Copyright, Licence, Warranty, Contact, Credits

All rights, including the copyright, for the Blender IA/IGS export extension *Bigex* are owned by HenningBR218.

(C) HenningBR218 2009-2011

The Blender IA/IGS export extension *Bigex* is subject to the GNU Public Licence, version 3 (GPL3) with the restrictions for commercial use as listed below. The full text of the GPL3 licence can be read on the net (see [\[7\]](#)).

The usage of the Blender IA/IGS export extension *Bigex* is allowed for creation of 3D models only, which are distributed without any form of compensation (Free ware). The usage for creation of 3D models which are not distributed as Free ware (Pay ware, Shareware, etc.) requires the express and written approval of the author (Contact: henningbr218 (at) b188 (dot) de).

Each 3D model, created with the Blender IA/IGS export extension *Bigex*, with a certain count of vertices gets a special marking. This mark appears later in the 3D model in Rail Simulator / Rail Works also. Licensed versions for commercial products are adding further customer specific marks. The evidence, that a certain 3D model was created with the Blender IA/IGS export extension *Bigex* is therefore identifiable.

The usage of *Bigex* is at the users own risk. The Author assumes no liability for damages, direct, indirect or consequential, to hardware, software or person resulting from the use of this script.

Any error reports, questions or suggestions for extension of *Bigex*, can be raised in the German language forum of **Rail-Sim.de** (see [\[8\]](#)) or in the English language forum of **UKTrainSim** (see [\[9\]](#)). The author can be contacted directly in exceptional cases via mail: henningbr218 (at) b188 (dot) de.

All rights of registered trademarks, branding marks and names, which are mentioned here in the manual, are and shall remain property of its current owner.

Many thanks for the help on manual translation, review and discussion to AndiS, Jeff Douglas (JADsHome) and karma99.

6 Introduction

The first Blender IA/IGS exporter by **steampsi** (see [\[1\]](#)) has got on in years. Like commonly known, it has some weak points and a few functions are missing completely. Especially the export of animations in the IA file format is supported incompletely. Furthermore it runs only with Blender versions which support Python 2.5 and is limited by this to Blender versions below 2.49.

Therefore, it was about time to write a new IA/IGS export plugin for Blender. Its name is Blender IA/IGS Exporter 2.0, or *Bigex* for short. It is completely new and developed from scratch. Like many other Blender plugins, *Bigex* is written in the programming language Python.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

7 About *Bigex*

7.1 Purpose of usage

The Blender IA/IGS export extension Bigex exports animated 3D objects created using Blender into the IA and IGS file formats. These file formats are required for importing 3D models and their animations into the simulation game software Rail Simulator (see [4]) or Rail Works (see [3]).

7.2 General

At export time, as many parameters as possible are exported from Blender into the IA/IGS files (see also tables in section 15.1 *Mapping of Blender data on the IA/IGS format*). All parameters not in Blender but required by the IA/IGS format, are set to meaningful default values.

Note: *Bigex* does **not** check if all data to be exported are matching the Rail Works specification (e.g. object names, shader names, etc.). Setting all parameter to get an IGS file which successfully exports to Rail Works is in the responsibility of the user.

Note: All statements in this manual in relation to Rail Simulator are valid for RailWorks too. Both simulators are able to read the IA and IGS files.

7.3 Feature overview

- Runs under Linux and Windows versions of Blender (32Bit and 64Bit)
- Exports animations into the IA and 3D object geometry data into the IGS file format
- Absolute, automated control on all values in the IA/IGS files by a powerful modification functionality for exported data, including regular expressions
- Supports the automated combining of multiple mesh objects into one
- Exports also optionally object hierarchies
- Creates optionally csv list files of all IA/IGS data for the import into a spreadsheet
- Allows setting of IA/IGS specific parameter via the Blender user interface
- Supports arbitrary lengths of names for objects, shaders, and textures
- Exports optionally all selected, visible objects only or all visible objects
- Supports multiple UV coordinate layer (UV Layer)
- Supports multiple texture levels (Texture Layer)
- Supports free assignment of UV Layer to Texture Layer
- Supports parallel usage of the old export plugin
- Supports Blender edge marker (SHARP, SEAM)
- Supports Blender modifier (EDGE SPLIT)
- Contains an analyzer mode for dump out of the contents of existing IA/IGS files into readable text files or as csv list for the import into a spreadsheet

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

7.4 History

2011-05-06 Bigex Version 2.1.71 released

Bugfix release:

- problem resolved which did prevent startup under Blender 2.48
- fixed issue with absolute positions of merged mesh object faces
- several minor bugs fixed

2011-02-14 Bigex Version 2.1.62 released

Major release:

- added IA-format (animation) export
- added user defined mesh object merging at export
- enhanced analysis of existing IGS and IA files
- added support for regular expressions for modify expressions
- freeware and customer specific model tagging improved
- several minor bugs fixed

2010-04-21 Bigex Version 2.0.158 released

First beta release:

- IGS-format (model geometry) export
- improved analysis of existing IGS and IA files
- added automated user defined IGS and IA file modification at export
- freeware and customer specific model tagging improved
- added support for Blender 2.49 / Python 2.6

2009-02-26 Bigex precursor IA/IGS Analyzer Version 1.0 released

Initial release.

- analysis of existing IGS and IA files
- content dump of IGS and IA files into a human readable text file

2006-06-01 Bigex pre-precursor Railsim IGS Export Preparator Version 1.0 released

Initial release.

- prerequisite check and preparation for RailSim (.igs) export

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

8 Installation

8.1 Supported program versions

Bigex runs under the following Blender / Python combinations:

1. Blender 2.48 (32/64Bit) / Python 2.5 (32/64Bit)
2. Blender 2.49 (32/64Bit) / Python 2.6 (32/64Bit)

All subversions of Python 2.5 and 2.6 can be used, in the 32bit or 64bit version.

Bigex runs under Blender 2.48. and 2.49 in the 32Bit or 64Bit Version.

It is planned to create a *Bigex* version running under Blender 2.5 / Python 3.0 once a stable, full featured Blender 2.5 version or higher is available.

Both programs are available for free in Linux and Windows versions.

Blender: see [\[2\]](#).

Python: see [\[5\]](#).

Note: *Bigex* version 2.1 does not run under Blender 2.5 / Python 3.0 or higher, because no stable, full featured Blender 2.5 version is available yet.

Bigex was tested under the following operating systems: Linux, Windows (XP, Vista, 7).

Together with an image processing program, like the free GIMP (see [\[6\]](#)), you have all tools required to create 3D objects for Rail Simulator (see [\[4\]](#)) and Rail Works (see [\[3\]](#)). The development environment formed by these tools runs under Linux as well as under Windows.

8.2 Required programs

To use *Bigex* the following programmes must be installed correctly:

- Blender, Version 2.48 or 2.49
- Python, Version 2.5 or 2.6
(depending on the Blender version, see section [8.1](#) for further guidance)

Note: *Bigex* was not tested for Blender versions lower than 2.48. There is a chance that *Bigex* will run on such versions, as long as they use Python 2.5.

8.3 Installing the *Bigex* scripts

The installation of the *Bigex* scripts is very simple. Just copy all files from the **Scripts** folder of the package into the Blender scripts folder.

Under Linux, this is: <home>/.blender/scripts

e.g.: ~/.blender/scripts

Under Windows there are two possibilities. Depending on which option was chosen when installing Blender, the Blender scripts are in different folders.

In case the choice was "**Install Blender for all users**" this script folder was used:

<Windows installation drive>:\Documents and Settings\<UserName>\Application Data\
Blender Foundation\Blender\blender\scripts\

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

E.g.: C:\Documents and Settings\UserName\Application Data\Blender Foundation\Blender\blender\scripts\

In case the choice was "**Install Blender this user only**" this script folder was used:

<programme drive letter>:\<program folder>\<Blender folder>\.blender\scripts

E.g.: C:\Programs\Blender\blender\scripts

Note: If the folder **.blender/scripts** does not exist, it can be created.

Note: If the **.blender/scripts** folder within the Blender installation directory is used (e.g.: C:\Programs\Blender\blender\scripts), the script is available only for the Blender version installed in that folder. The script is not available in other Blender versions, installed in parallel directories.

Note: From which folder Blender reads in scripts can be set in the Blender user settings (Window type: **i User Preferences**). If the field **Python Scripts** on the tab **File Paths** is left blank, both above mentioned standard folder are searched for scripts. If the user preferences are changed, the changes need to be saved (Menu *File => Save Default Settings Ctrl-U*). Otherwise all changes are lost.

After the next start of Blender the *Bigex* export script appears in the Blender menu (see 9.2 Starting the graphical user interface (GUI)) and can be used.

8.4 Blender example files

The installation package includes some example files. Blender file **Bigex_ExampleObject_1.blend** in folder **Bigex_ExampleObjects** contains complete examples for the usage of Rail Works shaders. The required texture files are in the subfolder **Textures**.

A 3D model for the commonly used shaders show all the Blender settings needed for the export into IGS. The example objects also show how many texture layer you need for each shader, and which parts of the texture image are used for which effect in Rail Simulator like e.g. transparency and gloss.

All examples are ready to be exported. To view the example objects in Rail Works the following steps need to be executed:

1. Unpack all files from folder **Bigex_ExampleObjects** to:
If **Rail Simulator** is used:
<Programs installation drive>:\<Program folder>\<RailSimulator folder>\
Source\HenningBR218\Addon\Bigex_ExampleObjects\

e.g.:

C:\Programs\RailSimulator\Source\HenningBR218\Addon\Bigex_ExampleObjects\

If **Rail Works** is used:

<Programs installation drive>:\<Programs folder>\<Steam folder>\
SteamApps\common\RailWorks\Source\HenningBR218\Addon\Bigex_ExampleObjects\

e.g.:

C:\Programs\Steam\SteamApps\common\RailWorks\Source\HenningBR218\Addon\Bigex_ExampleObjects\

Note: It is not possible to rename the **Source** folder or any of the subjacent folders. The exact folder structure and names are required by RailWorks respectively are fix defined in *Bigex_ExampleObject_1_exp.xml*. If any of the folder is renamed, the example object can not be imported into Rail Works any more.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

2. Load file *Bigex_ExampleObject_1_exp.xml* into the Asset Editor of Rail Works.
3. Export all example objects to Rail Works (Asset Editor button **Export**).

Optionally it is possible to export the IGS file first self. To do this, load the file ***Bigex_ExampleObject_1.blend*** and afterwards export one or all example objects via the Bigex user interface (see section 9.2 *Starting the graphical user interface (GUI)*).

A single shader example object can be exported by selecting it solely. If no object is selected then all example objects will be exported.

8.5 Further important files

The subfolder ***Bigex_ExampleObjects*** of the Bigex installation package contains four more files:

- IG_ExpModFile.txt
- IG_specific_IGExpModFile.txt
- IA_ExpModFile.txt
- IA_specific_IAExpModFile.txt.

Those contain examples for the search and replace expressions which are used to modify data at export time. Included are e.g. expressions to replace the short names for the most commonly used Rail Works shader into the long names, expected by Rail Works. One example expression is: *TrUpVFaceFlora=TrainUprightViewFacingFlora.fx*. This replacement is necessary due to the name length limitation in Blender. Otherwise those shader could not be used.

More example expressions can be found in the comment lines of those files.

So, if you want to export your own models it is strongly recommended to copy the both files ***IG_ExpModFile.txt*** and ***IG_specific_IGExpModFile.txt*** over into the folder, which is selected to create the IGS file in. Same is valid for the export of animations.

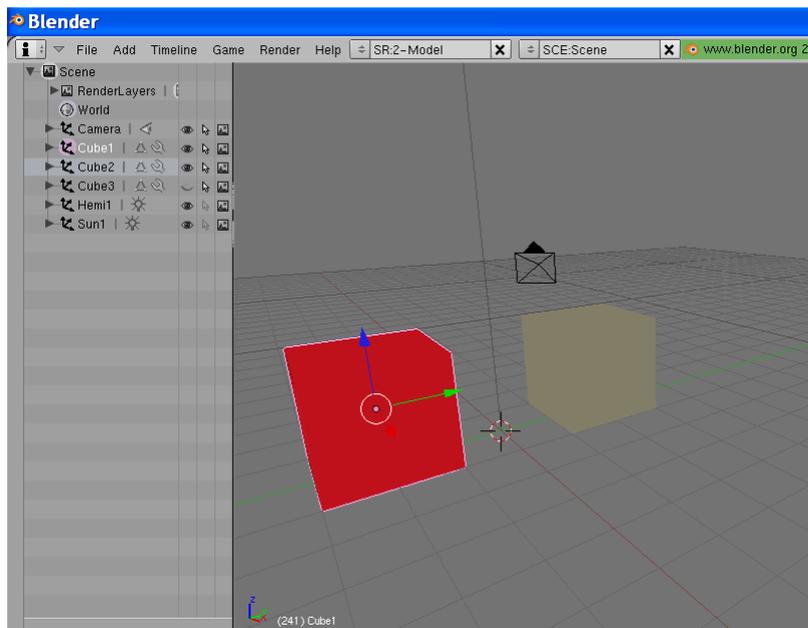
The file name with the model specific expressions should be adapted to the model file name. More on this later in section 12 *Data modification during export*.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

9 Usage

9.1 Exportable objects

As a general rule, only **visible** objects of type **Mesh** or **Empty** are exported. Objects, which are set on invisible, will not be exported. This means all objects that are marked with the greyed eye in the Blender *Outliner* window will not be exported.



Picture 9.1: Exported objects

In this example the selected mesh object *Cube1* is exported only. The not selected object *Cube2*, the hidden object *Cube3*, and all lamp and camera objects are not exported.

Illumination and camera objects are never exported in principle because the IGS file format does not know these objects.

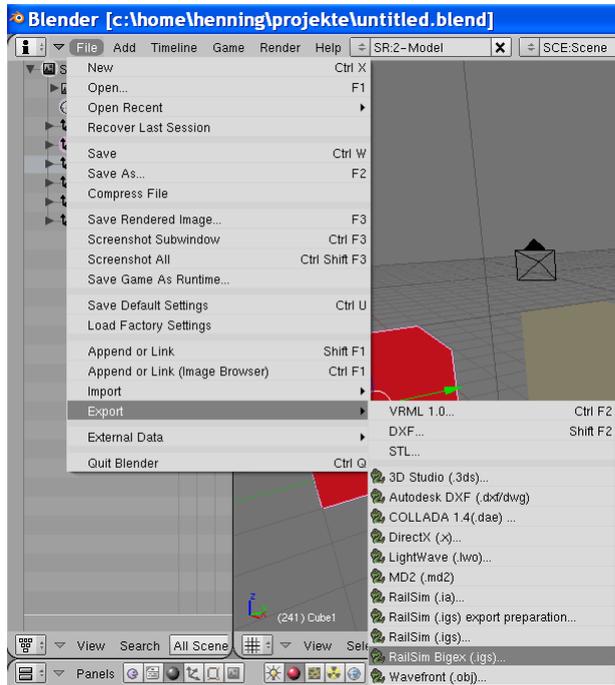
If exportable objects are selected, only these are exported. If no object is selected, all visible objects will be exported.

Illumination and camera objects are never exported in principle because the IGS file format does not have any data fields for those.

9.2 Starting the graphical user interface (GUI)

The export of the objects is done in two steps. In the first step the *Bigex* GUI, shown in picture 9.4, is started. From there the buttons *Export IGS* respectively *Export IA* starts the real export in the second step.

There are several ways to start the user interface. The simplest is using the Blender menus **File => Export => RailSim Bigex (IGS)** or **Scripts => Export => RailSim Bigex (IGS)** as shown on pictures 9.2 and 9.3. But of course, you can load the *Bigex* start script *RS_exp_igs_V21.py* respectively *RS_exp_ia_V21.py* into the text window just like any other script in Blender and then press ALT-P.

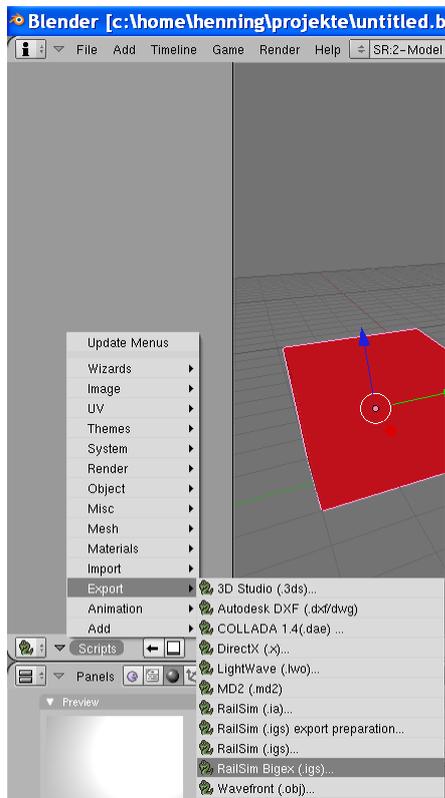


Picture 9.2: Starting the user interface 1

To start *Bigex* from a Blender menu, there are several alternatives.

The first is via the menu

File => Export => RailSim Bigex (IGS).



Picture 9.3: Starting the user interface 2

A second way is via the menu

Scripts => Export => RailSim Bigex (IGS).

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

9.3 Graphical user interface (GUI)



Picture 9.4: User interface

The user interface of *Bigex*.

On the right hand are the **input fields** for the Rail Works shader specific parameter. Indeed those are only used in very rare cases.

On the left are the blue export **option buttons** and at the bottom the grey **action buttons**.

On the left side of the *Bigex* user interface are the option buttons for the IA and IGS export: *Log File*, *Verbose*, *Obj Hierarchy*, *CSV Files* and *Std Exp Name*. In the bottom side section are the action buttons placed: *Export IA*, *Export IGS*, *Refresh* and *Exit*.

On the right side are the input fields for the very specific parameter which are used from some few Rail Works shader. All those input fields are exclusively used for Rail Works material and shader settings. Indeed are these specific parameter just for some very few Rail Works shader and thus used in very rare cases only.

Subsequent a brief description of all fields and buttons of the *Bigex* user interface.

9.3.1 Input fields and buttons of the *Bigex* user interface

Right below the header, in the field **Material**, the current Blender material is shown. This is the same than selected in Panel *Shading* => *Material* (F5). All input fields for the specific Rail Works shader parameter are belonging to this current material only.

In the drop down selection field **Viewer Facing** one of the predefined values can be selected, but it is used by some very few Rail Works shader only.

Below, in the field **RS** the current Render Stage can be set. The following eight Arg slider are belonging to this selected render stage. With those sliders the additional shader arguments can be entered. But just some very few Rail Works shaders are using additional arguments.

One Rail Works material can have up to eight render stages. Each of it can hold up to eight additional arguments. The numbering of the render stages as well as the arguments starts at zero. Thus the corresponding identifier are: RS0Arg0, RS0Arg1, ..., RS0Arg7, RS1Arg0, RS1Arg1, ... until RS7Arg7.

Almost all of the yet known Rail Works shader are using no additional arguments. Some few are using four arguments at maximum. Further information about the Rail Works shader are to be found in the developer documentation [13] and in the Rail Works Wiki [10].

In either case all preset values are written into the exported IGS file independent of the selected Rail Works shader is using those or not.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

All preset values in the *Bigex* user interface are saved in the *.blend* file together with the 3D model and are thus available again when the *.blend* file is loaded next time.

Note: Which values from Blender are written into which data fields of the IA/IGS file is listed in the tables in section 15.1 Mapping of Blender data on the IA/IGS format.

In the mid of the left side are the blue buttons for the export options. With the upper most option **Log File** logging of the console window output into a file is enabled. Enabling the option **Verbose** dumps out more details than shown in the console window. More details regarding Log-files can be found in the following sections.

If the option **Obj Hierarchy** is selected all parent-child relations (object hierarchy) is exported into the IGS file.

The option **CSV Files** controls if additional files shall be created containing comma separated values (csv) with all attributes of all exported objects. Such text files can afterwards very easily be imported and analyzed in a spreadsheet program like Openoffice Calc, LibreOffice calc or Microsoft Excel.

If the export is started with option *Standard Export File Name (Std Exp Name)* enabled, the name of the currently opened Blender model file with the extension *.ia* respectively *.igs* is used for the exported IA/IGS file. In this case no file selector window is displayed and the export starts immediately.

Existing IA/IGS files are replaced without asking for confirmation. To increase safety against unintended overwriting of an existing *.igs* file, an "_exp" is added to the file name.

Example: If your current Blender file is *MyEngine.blend*, your IGS file name with the model geometry would be *MyEngine_exp.igs* resp. *MyEngine_exp.ia* for the IA file with the animation.

If the option *Standard Export File Name (Std Exp Name)* is disabled when starting the export, a file selector window shows up as first step of the export. The name of the IA resp. IGS file to be exported can be entered. This chosen resp. entered file name is suggested again in the file selector window at the next export.

For easement of selection of existing file names in the file selector window *Bigex* adjusts the chosen file name to match the desired export file name.

Example: The IGS file *MyModel.igs* shall be generated. For this just one of the following file names need to be chosen in the file selector window:

- *MyModel_exp.ia*
- *MyModel_exp.igs*
- *MyModel_IGSexp.log*
- *MyModel_IGSExpModFile.txt*
- *MyModel.blend*

The adjustment on the desired file name *MyModel_exp.igs* is made by *Bigex* automatically.

Right after confirming with the button *Export IGS File* the export is starting.

Action buttons are found at the bottom of the user interface. The **Exit** button closes the *Bigex* user interface. The **Refresh** button will update the current material if *Bigex* does not do this automatically. Another option is to move the mouse pointer over a few other sub-windows of Blender. This will trigger the automatic user interface update function of *Bigex*.

The **Export IA** resp. **Export IGS** button starts the real export process.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

9.4 Export process

After clicking on the **Export IA** resp. the **Export IGS** button, *Bigex* starts mapping of objects and parameter from the Blender internal data structures into the IA/IGS data structures. Depending on the number of objects, vertices, polygons and materials as well as the PC hardware this is running on, this process can take seconds, minutes or hours.

During the export process *Bigex* works on copies of the Blender objects only. Neither objects nor parameters of the Blender data structure are modified!

The progress of the export can be monitored on the bar on the top of the Blender window, permitting a rough guide to how long the export will take.

The console window shows important warning and error messages throughout the export process. If the **Log File** option button on the *Bigex* user interface is selected all these information, plus some more, are written into a log file. This is created in the same folder than the IA/IGS file and has the same name as the *.ial.igs* file but with the extension of **_IAexp.log** resp. **_IGSexp.log**.

Example: With the opened Blender file *MyLoco.blend*, the resulting IGS export log file name would be *MyLoco_IGSexp.log*, respective *MyLoco_IAexp.log* for the IA export log file.

This log file lists in great detail and in readable format what is written into the binary IA/IGS file.

Note: For complex objects to be exported, the log file can become quite large. It can get a size of more than 100MB!

If this behaviour is not desired, the option button **Verbose** and, if required, additionally the button **Log File** need to be deselected.

When exporting *Bigex* does **not** check if **all** required data are supplied, parameter are set correctly and names are Rail Works compliant.

Note: It is the responsibility of the user that all data field in the IA/IGS file, like object names, shader names, shader settings, UV-layer, etc. are set properly to keep Rail Works specifications. This is precondition for a successful import of the generated IA/IGS file into Rail Works.

Bigex allows the user to set any value for all data fields of the IA/IGS file. There are no limitations.

Faulty set resp. missing values usually lead to crashes of the Asset Editor or termination of the import of the IA/IGS file into Rail Works (with a corresponding error message). Solution for such cases is the cross-check of all values in Blender which are exported regarding correctness and completeness. Same is valid for the search and replace expressions in the files with the modification instructions.

However, warning messages are created to hint on possible root causes of incompatibilities in the exported IA/IGS file with the Rail Works *Asset Editor*.

If any parameters are missing, for example ambient light colour, sensible default values will be exported by *Bigex*. If important information, like the name of a texture file is missing, warnings are issued and the object will be exported partially or not at all.

Therefore if whole objects or parts of them are not exported, it is recommended to check the log file for warnings and error messages for those objects.

If there are no major errors in the Blender objects, an IGS file is written. Otherwise, the export process terminates prematurely with an error message.

Finally *Bigex* writes a status report and a set of statistical values to the console window and the log file. They tell you how much time each of the export steps took and how many objects, vertices, polygons and materials were written into the IA/IGS file.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Note: Minimising the console window reduces the total time the export process takes. Printing and scrolling this window significantly adds to the total duration of the export process.

To see messages already dumped during the export, open the log file in an external editor and reload it from time to time.

In some cases the import of the IA/IGS file into the asset editor of Rail Works fails. In other cases the conversion in the asset editor into the *.GeoPcDx* file terminates with an error message. To discover the cause it is a good idea to have a closer look at the warning messages in the export log file. If all values in Blender are set in a way that *Bigex* generates no warnings, the whole import path into Rail Works should run error free.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

10 Creating objects for the IA/IGS export in Blender

10.1 Object types

In basic only two types of object are exported by *Bigex*. These are Blender *Mesh* objects, containing the geometry data of a 3D model and *Empty* objects, which are used amongst others for grouping of the mesh objects. All other Blender objects like splines, cameras and lamps are not exported.

10.2 Object orientation

The coordinate systems of Blender and Rail Works are not the same. Rail Works uses a left handed coordinate system (positive values point: X right, Y up, Z forward) and Blender is using a right handed coordinate system (positive values point: X right, Y forward, Z up). The conversion between these is done automatically by *Bigex* at export time.

This means, if for example you build a loco in Blender in a way that the X axis (red arrow) points to the right, the Y axis (green arrow) points in driving direction and the Z axis (blue arrow) points upward, then the loco has the correct orientation in the Rail Works.

10.3 Object hierarchies

In basic the IGS format supports object hierarchies with unlimited depth. Also *Bigex* supports the export of Blender object hierarchies into the IGS file format with an unlimited depth.

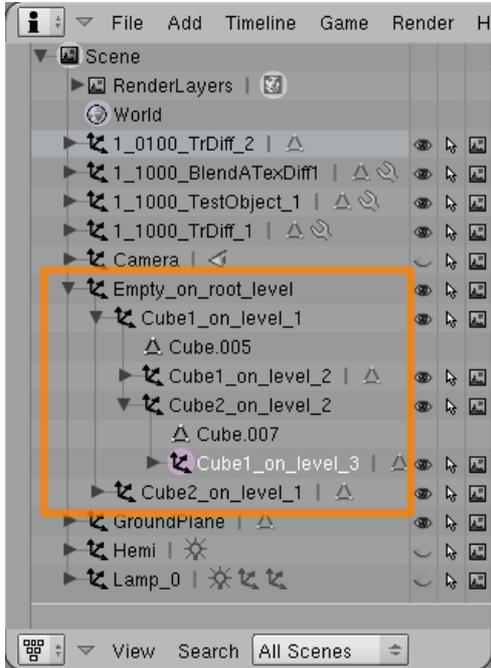
Admittedly object in IGF format can hold only 24 child objects at maximum. If a Blender object has more than 24 child objects, then for each further child object a corresponding warning message is issued and it is exported on the top object hierarchy level (root).

Object hierarchies are not always required. Thus the export can optionally be disabled by deselecting the button **Obj Hierarchy** on the *Bigex* user interface. In that case all objects are exported at the top level (root) and thus there is no object hierarchy existing any more.

If the conversion tool *ConvertToGEO.exe*, used by Rail Works Asset Editor to convert the IGS file into the GeoPcDx format, terminates with the error message "*Instancing not supported*", it is recommended to export the IGS file again without object hierarchies.

Picture 10.1 shows an example for object hierarchies in Blender.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------



Picture 10.1: Object hierarchies
Object hierarchies in Blender with a depth of four.

The example in picture 10.1 shows an object hierarchy at which the object *Cube1_on_level_3* is on the third level below the root level. The whole hierarchy structure is exported into the IGS file if the option **Obj Hierarchy** is enabled.

If just some selected objects shall be exported, it is tried to keep the remaining hierarchy intact as far as possible.

10.4 Known issues at import into Rail Works

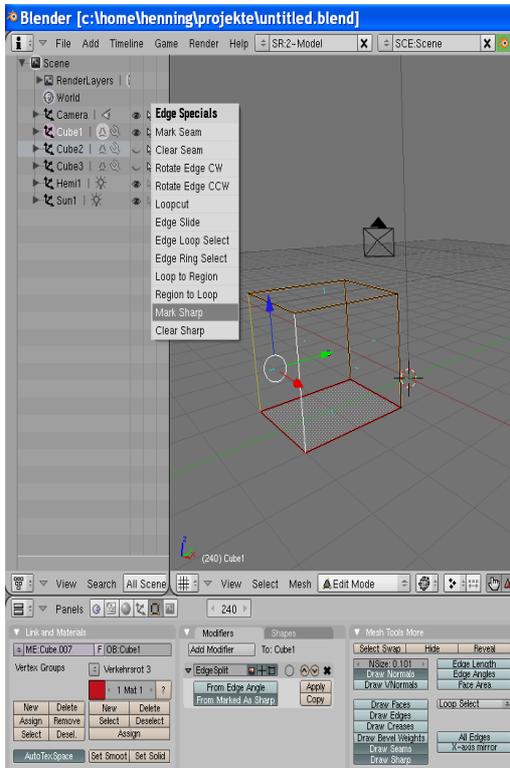
One known issue of the conversion tool *ConvertToGEO.exe* of the Asset Editor of Rail Works is maximum number of 256 mesh objects per IGS file. If the IGS file contains more than this maximum, the conversion tool terminates with a corresponding error message. The only solution for this is to merge mesh objects to get the number below the maximum.

Merging mesh objects can be done either in Blender itself or by using *Bigex*. But the former is irreversible and the feasibility is lost to copy single mesh objects later for a e.g. mirrored or scaled use.

A more smart method would be merging objects at export time. This remains the 3D model in Blender unchanged. More on this later in section 12.4.1 *Merging of mesh objects*.

10.5 Edge marker

Blender offers some very useful edge markers. Marked edges are highlighted in different colours. On the *Mesh Tools More* tab in Blender you determine which types of edges are shown in colours. For the marker and modifier shown below, these are *Draw Seams* and *Draw Sharp*. Which edges will be highlighted can be adjusted in edit mode on the *Editing (F9)* menu and the *Mesh Tools More* tab - e.g. the edge marker *Draw Seams* and *Draw Sharp*.



Picture 10.2: Edge Marker

In this example, in menu *Mesh Tools More* the face normals (little blue lines on the visible side of the face), the **Seam** (orange) and **Sharp** (red) edge markers are selected to be visible.

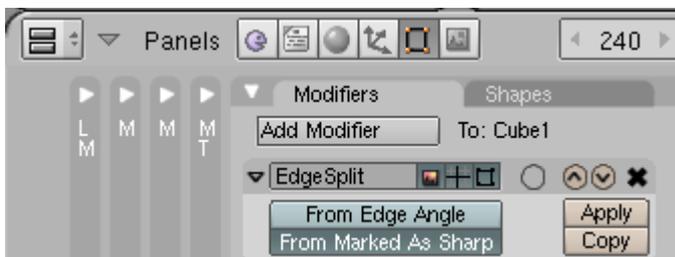
Selected edges can be marked with the *Edge Specials* pop-up menu (CTRL-E key) as **Seam** and **Sharp**.

As a help for flexible UV Layer creation (unwrapping), Blender provides the edge marker **Seam**. This marker (orange edges) is supported by *Bigex*. This means additional vertices are generated as required for the export of all UV coordinates.

Another very useful edge marker is **Sharp** (red edges). This one is used together with the modifier **EdgeSplit** (see section 10.6) to obtain hard edges on an object that is set to smooth (*Set Smooth*). For example at a closed half of a sphere. The **EdgeSplit** modifier should be set to *From Marked as Sharp*. *From Edge Angle* should be deselected in this case.

10.6 Modifier

Nearly all modifiers are supported by *Bigex*. During export, all modifiers for an object are applied to a copy of the original object. This copied object is then exported and deleted afterwards. The original object remains as is with all its modifiers.



Picture 10.3: Modifier

Example: The very useful modifier **EdgeSplit**.

Refer also to section 10.5 Edge marker.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

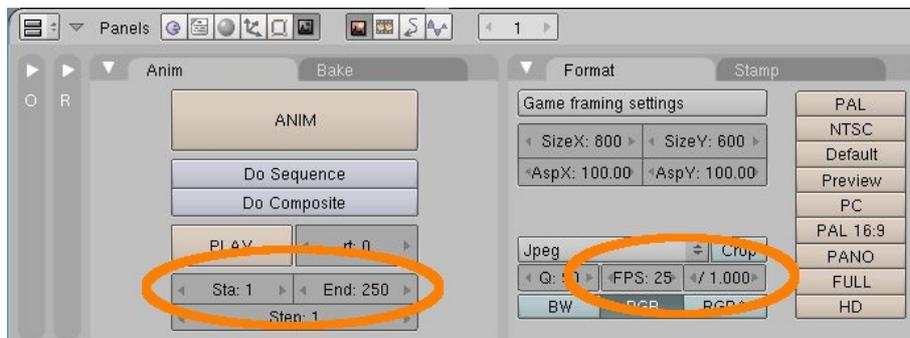
10.7 Animations

In basic all animations created in Blender can be exported into the IA format by *Bigex*. This is regardless of which Blender animation technique was used: Armatures/Bones, Constraints or IPO Keys.

For which objects an animation is exported follows the object selection of the IGS export. If any objects are selected in Blender, the animation for those will be exported. If no object is selected all animations for all objects will be exported. For static objects, means objects which do not move, a static animation will be written into the IA file.

At the animation export all frames are considered as set on panel *Scene (F10)*, tab *Anim* of Blender. In this process the frame with the number set in input field **Sta** is the first exported frame and the one in input field **End** the last.

The frame rate for the IA export is calculated by **Multiplication** of two values from tab *Format*. The first one is from the input field **FPS** and the second one from *Frames per second base*, just right of the **FPS** input field. A multiplication is required due to the value range of the Blender input field FPS is limited to 120fps, but Rail Works is able to process larger values.



Picture 10.4: Animation parameters

Start and end frame and the frame rate.

The export in the IA file format is started by clicking the button **Export IA** on the Bigex user interface.

If the option *Standard Export File Name (Std Exp Name)* on the *Bigex* user interface is disabled a file selector window shows up as first step of the export. The path and name for the IA file to be exported can be chosen. If the **Std Exp Name** option is enabled the name of the currently opened Blender model file with the extension *.ia* is used. In this case no file selector window is displayed.

Existing IA files are replaced without asking for confirmation. To increase safety against unintended overwriting of an existing *.ia* file, an "_exp" is added to the file name.

If a common or model specific text file with search and replace expressions exists in the chosen IA file target folder, the IA export data will be modified accordingly. More on this later in section 12 Data modification during export.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

11 UV coordinates and Texture layer

11.1 Overview

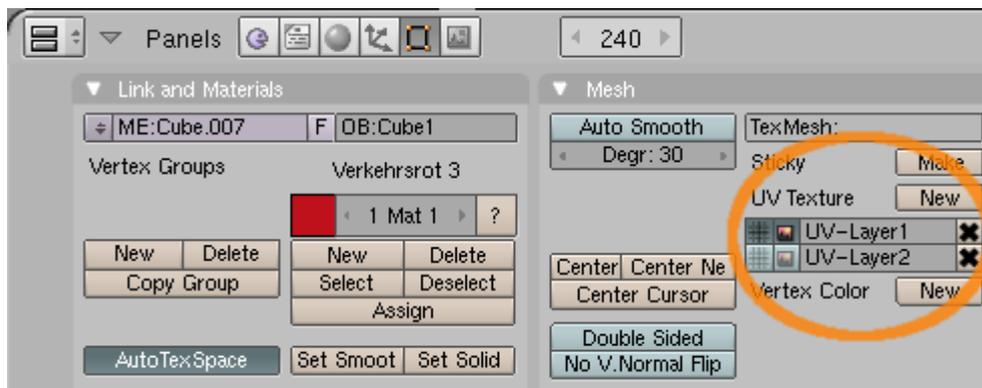
To texture a 3D model, you need **UV coordinates**, a **texture image** and a **shader name**.

This section overview explains these terms in brief. The following three chapters describe the details and the usage more in depth.

UV coordinates define for each vertex the position on the texture it is mapped to. Because a **texture image** has two dimensions, only one pair of coordinates are required. These are the X and Y position on the **texture image**, which is mapped to the relevant vertex. For preventing confusion with the X, Y and Z coordinates of the vertex's geometry data, the coordinates on the texture image are called U and V.

A list containing all the UV coordinates for all vertices of one mesh object is called the **UV coordinate layer**, or short UV Layer. In Blender it is possible to set a name for each UV Layer for easier identification. Each mesh object in Blender can have multiple UV Layers each with their own unique names.

Unfortunately Blender calls this UV Layer "**UV Texture**". This should not be confused with the term **texture layers**, described below. In this document, we will stick with the term "**UV Layer**" for a set of UV coordinates of a mesh.



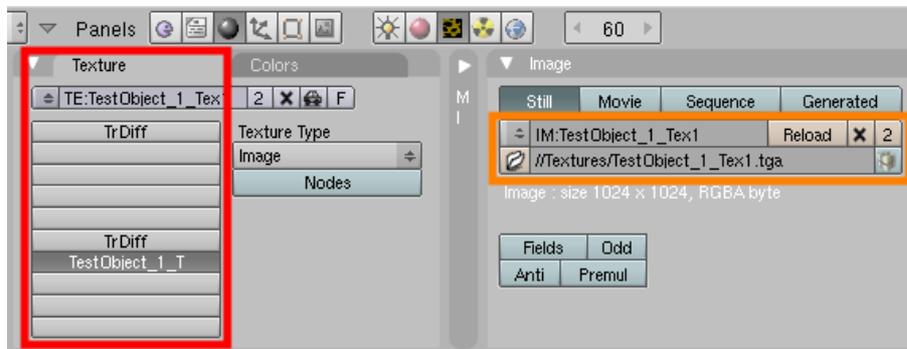
Picture 11.1:
UV Texture

Two UV Layer with a set of UV coordinates each for a Blender mesh object.

A **texture image** in Blender is a reference to a path and file name pointing to a texture image file. A **texture image** can be specified via a **texture layer** for Blender materials. Each texture layer can hold one **texture image** reference. Furthermore a texture layer holds information about what **UV layer** is to be used to map the **texture image** on the mesh vertices. Of course it is possible to have multiple texture layers for a material, each with it's own unique name.

A texture layer can be unlimited be reused within the same Blender material as well as for other materials.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------



Picture 11.2: Texture Layer

The Texture *Layer* stack of Blender in the *Texture Menu* (red box) containing three defined layer. The active layer number seven *TestObject_1_T* has the texture image *TestObject_1_Tex1.tga* assigned (orange box).

Important note: Images, selected in the *UV/Image Editor* window of Blender at UV coordinate creation, are totally independent from the *texture images* of the *texture layer*! Texture images set in the *UV/Image Editor* window are exclusively used in Blenders 3D windows. For the export into the IGS files and the later use in Rail Works, as well as for rendered pictures in Blender, the *texture images* specified in the *texture layers* are used.

A shader name must be specified for Rail Works to display the 3D object. The available Rail Works shader, as well as their exact names and parameters, are thus fix specified by Rail Works. Depending of the chosen shader it requires one, two or three *texture layers*, each one with a *texture image* assigned. The Rail Works shader names are passed to Blender via one of the texture layer name (see section 11.3 Texture layer).

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

11.2 UV coordinate Layer

UV Layers are lists with two dimensional coordinates of points on a texture image, to which the vertices of a mesh are mapped on. UV coordinates are later on used in Rail Works to map texture images exactly on the faces of 3D objects.

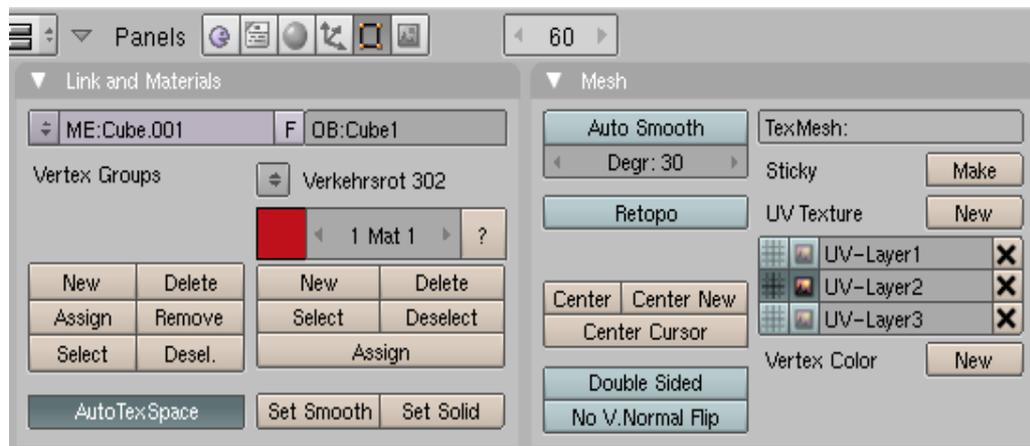
Usually one UV layer is sufficient. In exceptional cases it can be necessary to have a second or third UV Layer for a mesh. This may be the case for a second texture - for example normal mapping data or a gloss texture, which shall be used for a few faces only or lays differently on the mesh.

To create UV coordinates in Blender, select those faces in edit mode for which you want to set the UV coordinates. Then apply the "unwrap" method of Blender on these faces by pressing the **U**-key. This brings up the *UV Calculation* menu, which offers a list of unwrapping methods. Simply experiment with the available alternatives to see which one delivers the best results for your needs.

After this step, the UV coordinates of the selected vertices are added to the active UV layer. If there was none before, a new one is created. The new UV layer can be found, with a selected mesh object, under *Editing (F9) => tab Mesh*, directly underneath *UV Texture*.

Further UV layers can be created using the *New* button. Note that the UV coordinates are always added to the **active** UV layer.

Names for UV layers can be chosen arbitrarily. In the example below on picture 11.3 *UV-Layer 1* and *UV-Layer3* are inactive while *UV-Layer 2* is activated.



Picture 11.3:
UV Layer

UV layer, in Blender to be found in the *Mesh* menu under the term *UV Texture*.

11.3 Texture layer

The texture layers in Blender are used to hold the parameters of the single render stages for the Rail Works shader. Texture layers are part of a Blender material.

Each mesh object and each face in Blender must have a material assigned to it, otherwise it cannot be exported to IGS. The assigned material prescribes the way the object is displayed in Rail Works later, i.e., the visual properties of the object.

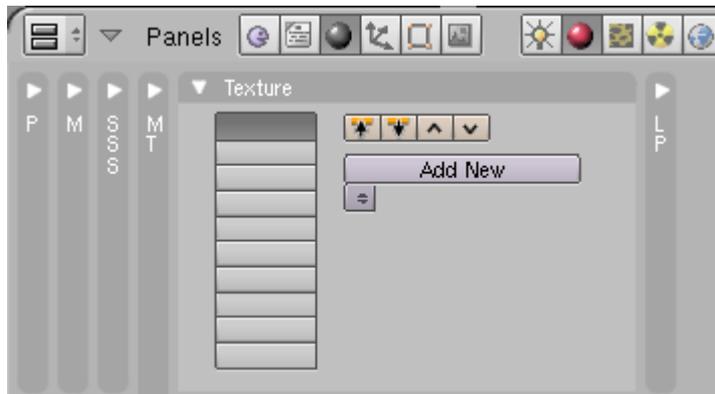
In Blender, there are many parameters to define these properties. However, the IGS format can hold a small subset of them only.

One of the most important properties are *texture layers*. These define the shader which is used by Rail Works later, as well as the *texture images* to be used for the render stages of that shader.

For each texture layer, one file name can be specified, which points to the image for the texture. The images should be in *.tga* or *.png* format because those both formats can hold alpha channel data. The *.tga* format is preferred and should be used.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

The material panel (*Shading (F5) => Materials Button => tab: Texture*) contains a stack of texture layers for each material. To set one of the texture layers, simply choose an existing one or create a new one by selecting *Add New*. Afterwards set *Texture Type* to *Image* and select your texture file with the file selector on the *Image* tab like shown on picture 11.2.



Picture 11.4: Texture Layer Stack

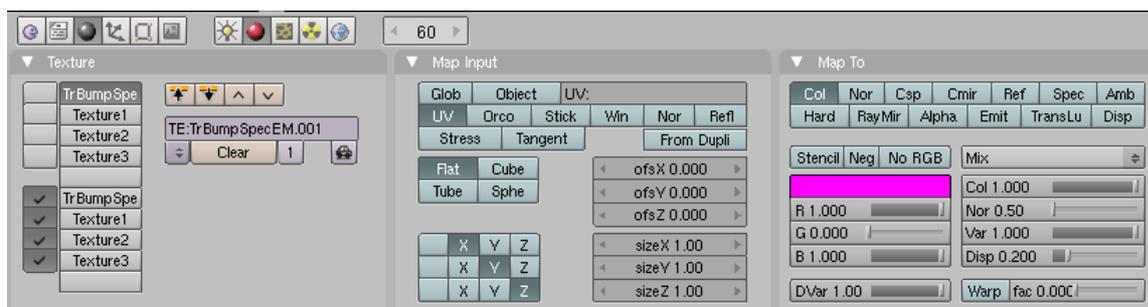
The empty *Texture Layer* stack in Blender, this time in the *Material* menu.

As mentioned before, Rail Works requires it's own, specific shader names to display the 3D models correctly. Thus a specific Blender texture layer is used to set the shader name for the later IGS export.

More information on Rail Works shader are in section 11.5 Shader.

Important note: The shader name for a material which is used for the export into the IGS format is specified via the name of the **first active** texture channel.

Important note: Due the name length in Blender being limited some shader names need to be set using a short form. During the export, those are replaced by the full Rail Works shader names (see 12 Data modification during export).



Picture 11.5: Active Texture Layer

In the example in picture 11.5 the sixth *texture layer* of the list is the first active one. This one defines the shader name *TrBumpSpecEM* in this example. The following three active texture layers define the other texture files required by this shader.

Exclusively the name of the first active texture layer is used. **No** further settings are used for export into the IGS file. By this, there is a clean separation between shader name and texture layer definitions. For this reason it is required only to define one texture layer with a shader name for each Rail Works shader. This one can be reused as often as needed for other materials.

The real texture layer specifying the texture images for the chosen shader are defined in one of the following entries. Depending upon how many the chosen shader expects, the corresponding number of texture layers need to be entered and activated. Due to the names of these texture layers are not used for the export, but only the texture image references and some other settings, all of that texture layer can be reused unlimited for other materials too.

Important note: Empty and/or inactive texture layer entries (i.e., without a tick mark) will be ignored by *Bigex*.

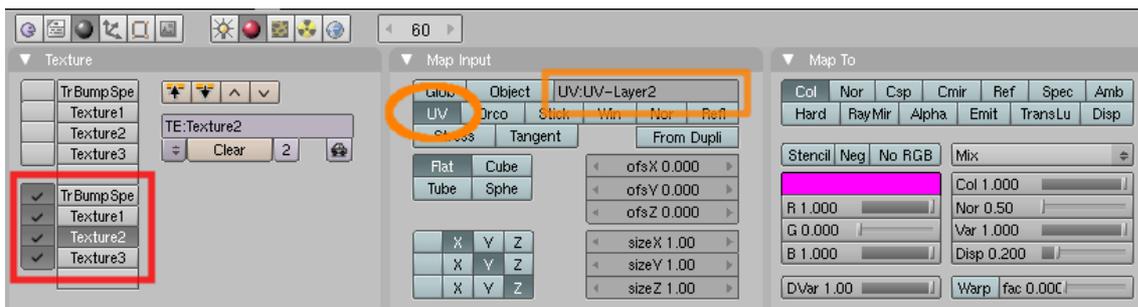
This means, the next active texture layer, following the shader name texture layer, is for the first render stage of the shader, the next active one for the second, and so on. All active texture layers will be exported, independent of what number of the layer the shader expects.

Note: *Texture layer* are called **Render Stages** in Rail Works. Both terms name the identical attribute of shaders. All Blender Texture Layer, except the first for the shader name, are exported one to one into the IGS file and by this mapped onto the Rail Works **Render Stages**.

The following parameters of the single texture layer are used for the export into the IGS file format:

Tab **Map Input**, Button **UV**: Must always be selected.

Tab **Map Input**, Text field **UV**: If there are multiple UV layers, the name of the intended UV layer must be specified here. If this field is left blank, then the UV coordinates from the active UV layer are used. Usually there is only one UV layer required and this field remains blank.



Picture 11.6: Render Stages

The example in picture 11.6 shows four activated texture layers (red box). For the second texture pass of shader *TrBumpSpecEM* it is specified that the UV coordinates from the UV layer with the name *UV-Layer2* (orange box) shall be used.

Some further parameters of this texture layer are exported into the IGS file too. For a detailed list refer to the table in section 15.1.1 Blender Input fields for the IGS format.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

11.4 Backward compatibility with old IGS export plugin

If you would like to have the choice to export your 3D model with the old IA/IGS exporter plugin instead there exists an easy solution. This may be required as long as *Bigex* does not support the IA format and you may want to export animated objects.

The solution consists of two separate texture layer groups. One for the old IGS export plugin and one for *Bigex*. Both exporters expect the shader name as the name of a specific texture layer. But in Blender it is not possible to have two texture layers with the same name but different settings.

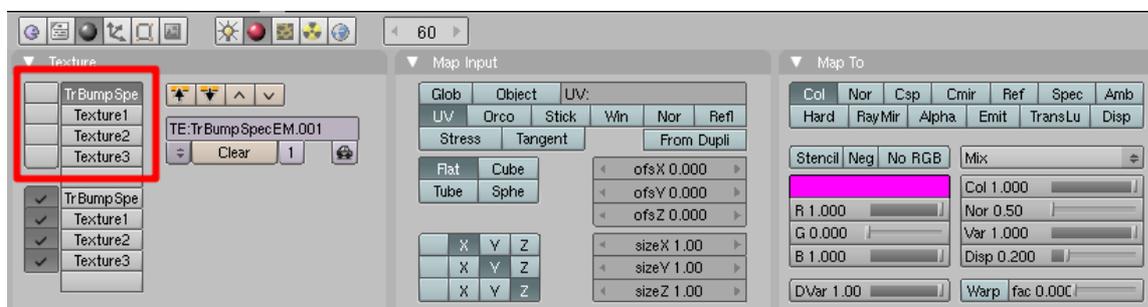
There is however an easy workaround for this problem. The old IGS export plugin for Blender (see [1]) expects the shader name in the **very first** entry of the texture layer list. Fortunately it accepts the short form of the Rail Works shader names, but also accepts an extension of the shader name of the form ".001, .002, ...". By this extension of the shader name / texture layer name it is possible to have a clear separation between both texture layers with the same shader names.

Directly following, starting from the second entry of the texture layer list the old IGS exporter plugin expects one or more further texture layer entries, if applicable. For all following not required texture layers the old exporter reports a warning message, but ignores it. In difference to *Bigex* the old exporter ignores if the texture layer is activated or not.

The texture layer group with the red border in picture 11.7 shows the expected texture layer for the old exporter.

The activated group underneath is the texture layer *Bigex* expects.

Like mentioned before, the old exporter runs under Blender 2.48 and Python 2.5 only. If you have this combination running, then you can export by choice with both exporters **without** having to change anything.



Picture 11.7: Settings for export with both exporter plugins.

Note: All texture layers, except those for the shader names, can be used in both groups one or more times. The old exporter uses just the texture image file name in difference to *Bigex*, which uses further settings.
(see section 15.1.1 Blender Input fields for the IGS format)

Another issue is to taken into account for the old exporter. The expansion of the short shader names into the long names is fix build into the code. The exact shader names from the documentation have to be used. Due to the shader names are hard coded, the old exporter doesn't know neither the new Rail Works shader nor the new parameter of the changed shader. By this, all those shader can not be used with the old exporter.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

11.5 Shader

The table below lists almost all of yet known Rail Works shaders.

Note: Because the count of the shaders, as well as their parameter, can be changed any time by the Rail Works developer team, by means the table below is not complete and up to date.

The first column shows the official, full written Rail Works shader name. The second column lists the short form for use in Blender if contained in the modify expression file.

The following columns contain, as far as known, a brief shader description, count of texture layer and a brief description about the single textures / texture layer / render stages.

Further and more up to date information about Rail Works shaders can be found in the *Rail Works Developer Package* documentation (see [\[12\]](#)) and in RS WIKI (see [\[10\]](#)).

The shader are split into two groups: Non-fx shader and fx shader.

Texture images come in two forms and differ in colour depth:

- RGB images with 24Bit per pixel (8Bit red, 8Bit green, 8Bit blue)
- RGBA images with 24Bit RGB image plus 8Bit Alpha channel (Grey scale) with a total of 32Bit per pixel.

Non-fx Shader					
Shader Name (Non-fx)	Short Name	Description	Texture 1	Texture 2	Texture 3
AddAlphaDiff	AddAlphaDiff	No texture, additive vertex alpha with diffuse colour.			
AddATex	AddATex	Texture mapped, no lighting applied, using additive alpha from texture's alpha channel	RGB: Colour A: Transp.		
AddATexAlphaDiff		Texture mapped, with diffuse colour, using additive alpha from texture's alpha channel combined with vertex alpha			
BlendATexDiff	BlendATexDiff	Texture mapped, with diffuse colour, using additive alpha from texture's alpha channel	RGB: Colour A: Transp.		
AddDiffuse		No texture, with diffuse colour, using additive alpha			
AddTex	AddTex	Texture mapped, no lighting applied, using additive alpha	RGB: Colour		
AddTexAlphaDiff		Texture mapped, with diffuse colour, with additive vertex alpha			
AddTexDiff		Texture mapped, with diffuse colour, using additive alpha			
BlendAlphaDiff		No texture, vertex alpha blending with diffuse colour			
BlendATex		Texture mapped, no lighting applied, using alpha blending from texture's alpha channel			

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Non-fx Shader					
Shader Name (Non-fx)	Short Name	Description	Texture 1	Texture 2	Texture 3
BlendATexAlphaDiff		Texture mapped, with diffuse colour, using alpha blending from texture's alpha channel combined with vertex alpha			
BlendATexDiff		Texture mapped, with diffuse colour, using alpha blending from texture's alpha channel			
BlendATexDiffTrans		Texture mapped, diffuse colour, alpha blending from texture's alpha channel, pixels with alpha=0 are transparent (e.g. alphaed fences).			
BlendTexAlphaDiff		Texture mapped, with diffuse colour, with vertex alpha blending			
BridgeSplit		Not drawn. Use to define areas where track crosses over itself.			
Diffuse		No texture, just diffuse colour			
DualAddATexDiffDestBlend		Dual textured, diffuse colour, first pass additive, and second pass blended alpha with the alpha of the first texture (e.g. puddles).			
DualBlendATexDiffAdd		Dual textured, with diffuse colour, using alpha blending for first pass and additive alpha for second pass			
DualTexDiffAdd		Dual textured, with diffuse colour, using additive alpha for second texture			
DualTexDiffAddWithLightIntens		Add second pass to first pass, brightness of second pass affected by light maps if used			
DualTexDiffAddWithoutLightIntens		Add second pass to first pass, brightness of second pass not affected by light maps if used			
DualTexDiffInvisibleStencilBlend		Dual textured, with diffuse colour, first pass invisible, second pass alphaed using alpha of first pass texture			
DualTexDiffStencilAdd		Dual textured, with diffuse colour, using additive alpha for second texture only where first texture has solid alpha			
DualTexDiffStencilBlend		Dual textured, with diffuse colour, using blended alpha for second texture only where first texture has solid alpha			
DualTexDiffTAlpha		Dual textured, with diffuse colour, using second texture's alpha channel to blend between textures			

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Non-fx Shader					
Shader Name (Non-fx)	Short Name	Description	Texture 1	Texture 2	Texture 3
DualTexDiffTrans		Dual textured, with diffuse colour, using second texture's transparency			
DualTexDiffVAlpha		Dual textured, with diffuse colour, using vertex alpha to blend between textures			
EmbossBumpmap		Bump map for Train 2 prototype or something like that			
Invisible		Nothing is drawn - use for invisible collision barriers			
Tex	Tex	Texture mapped, no lighting applied	RGB: Colour		
TexDiff	TexDiff	Texture mapped with single texture, diffuse colour applied	RGB: Colour		
TripleGlossMap		Triple texture, 2nd pass contains gloss map in alpha channel, 3rd pass (reflection) texture drawn additively			
TripleGlossMapWithLightIntens		Triple texture, 2nd pass alpha channel gloss map, 3rd pass drawn additively affected by light maps if used			
TripleGlossMapWithoutLightIntens		Triple texture, 2nd pass alpha channel gloss map, 3rd pass drawn additively not affected by light maps if used			
TripleTexDiffAddAdd		Triple textured, 2nd and 3rd passes are drawn additively			
TripleTexDiffTAlpha		Triple textured, with diffuse colour, using each texture's alpha channels to blend between each pair of passes			
TripleTexDiffTAlphaVAlpha		Triple textured, with diffuse colour, pass 2 uses texture alpha for blending, pass 3 uses vertex alpha for blending			
TripleTexDiffVAlpha		Triple textured, with diffuse colour, using same vertex alpha to blend between each pair of passes			
TripleTexDiffVAlphaTAlpha		Triple textured, with diffuse colour, pass 2 uses vertex alpha for blending, pass 3 uses texture alpha for blending			

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

fx-Shader					
Shader Name (fx)	Short Name	Description	Texture 1	Texture 2	Texture 3
TrainEnv.fx	TrEnv		RGB: Colour	RGB: Dummy	
LoftTexDiff.fx	LoftTexDiff		RGB: Colour		
LoftTexDiffTrans.fx	LoftTexDiffTr		RGB: Colour A: Transp.		
LoftBump.fx		Diffuse texture and normal map			
LoftBumpAlpha.fx		Diffuse texture with alpha and normal map			
LoftBumpTrans.fx		Diffuse texture with 1-bit alpha and normal map			
SkinAmbient.fx		Single colour skinned			
SkinDiffuse.fx	Skin	Textured skinned.	RGB: Colour A: Transp.		
SkinGloss.fx		Textured, normal mapped, specular with gloss map, and skinned.	RGB: Colour	RGB: Normal Map	RGB: Gloss Map
SkinNormal.fx		Textured, normal mapped, specular and skinned.	RGB: Colour	RGB: Normal Map	
SkinSpecular.fx		Textured, specular and skinned.	RGB: Colour		
StencilShadow.fx	Shadow	Stencil shadow objects, material must begin with shadow_ to be detected	RGB: Colour		
TrainBasicObjectDiffuse.fx	TrDiff	Single texture, dynamic lighting.	RGB: Colour		
TrainBasicObjectSpecular.fx	TrSpec	Texture, colour modulated specular.	RGB: Colour A: Transp.	RGB: Spec color map	
TrainBumpEnv.fx		Textured, normal mapped, environment mapped.	RGB: Colour	RGB Normal Map	RGB: Dummy (Cubic Env)
TrainBumpEnvMask.fx		Textured, normal mapped, masked environment map.	RGB: Colour A: Env Mask	RGB: Normal map	RGB: Dummy (Cubic Env)
TrainBumpSpec.fx	TrBumpSpec	Textured, normal mapped, specular.	RGB: Colour A: Transp.		
TrainBumpSpecEnv.fx	TrBumpSpecEM	Textured, normal mapped, environment map and specular.	RGB: Colour	RGB Normal Map	RGB: Dummy (Cubic Env)
TrainBumpSpecEnvMask.fx		Textured, normal mapped, masked environment map and specular.	RGB: Colour A: Env & Spec Mask	RGB: Normal map	RGB: Dummy (Cubic Env)
TrainBumpSpecMask.fx		Textured, normal mapped, masked specular.	RGB: Colour A: Env Mask	RGB: Normal map	
TrainFlora.fx	TrFlora	Ambient lighting, single texture.	RGB: Colour	none	
TrainGlass.fx		Screen space refractive glass with normal map and diffuse.	RGB: Colour	RGB: Normal map	Back buffer copy

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

fx-Shader					
Shader Name (fx)	Short Name	Description	Texture 1	Texture 2	Texture 3
TrainLightMapWithDiffuse.fx	TrLightMap	Diffuse tex, light map, dynamic lighting.	RGB: Colour	RGB Light map	
TrainSkyDome.fx	Sky	Skydome	RGB: Colour	RGB: Dummy (Cubic Env)	
TrainSpecEnv.fx		Textured, vertex environment mapped with specular.	RGB: Colour	RGB: Dummy (Cubic Env)	
TrainSpecEnvMask.fx	TrSpecEM	Textured, masked vertex environment mapped with specular.	RGB: Colour A: Env & Spec Mask	RGB: Dummy (Cubic Env)	
TrainUprightViewFacingFlora.fx	TrUpVFaceFlora	Single texture, globally lit, upright view facing	RGB: Colour A: Transp.		
TrainVertexLit.fx		Diffuse tex, vertex lighting only.	RGB: Colour		
TrainVertexLitWithDiffuse.fx		Diffuse tex, vertex lighting, dynamic lighting.	RGB: Colour		
TrainViewFacingFlora.fx	TrVFaceFlora	Single texture, globally lit, view facing	RGB: Colour A: Transp.		
WaterCubeMap.fx	Water	Splash	RGB: Colour A: Transp.	RGB: Normal map	
SkinRESERVED1.fx		RESERVED FOR FUTURE USE			
SkinRESERVED2.fx		RESERVED FOR FUTURE USE			
SkinRESERVED3.fx		RESERVED FOR FUTURE USE			
TrainBumpEnv.fx		Textured, vertex environment mapped.	RGB: Colour	RGB: Dummy (Cubic Env)	
TrainBumpEnvMask.fx		Textured, masked vertex environment map.	RGB: Colour A: Env. Mask	RGB: Dummy (Cubic Env)	

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

12 Data modification during export

12.1 Basic functionality

Bigex has a very powerful built-in function for data manipulation during the export which allows the setting of **every** entry in the IA/IGS file to an arbitrary value. This **Export Data Modification** function, called **EDM** in the following, is working on base of search and replace expressions. These expressions are read from some user defined text files.

Basically EDM differentiates between modification expressions for object attributes and action expressions for modification of 3D model geometries. More details on both types of expression can be found in the following sections 12.3 Modification of object attributes and 12.4 Action expressions.

The *Bigex* EDM function supports regular expressions which enables the unlimited modification of data field values in the IA/IGS files. By this the absolute control about all data in the IA/IGS files is given which means the appearance and characteristics in the simulator itself. Additionally this enables the possibility for unlimited experiments with all shader parameters.

Most common usage of the object attribute modification is e.g. for replacing the short form of the Rail Works shader names into the full shader name or replacing the path and name of a texture file for dynamic numbering. Action expressions are used for e.g. merging several mesh objects into one mesh object. This is used for e.g. if more than 16 materials are required for one Blender mesh object or for exporting more than 256 mesh objects for one 3D model.

The Blender internal data structures remain unchanged. Only the exported data values are changed.

Basically all data can be manipulated which are exported into the IA/IGS format file. These are not only strings like shader and object names but also all integer and floating point number values. This for example enables the manipulation of the floating point numbers of the world colour or the positioning of objects in the IGS file.

Manipulating the data follows the search and replace principle. The built in EDM search & replace function also supports regular expressions which follow the **Python syntax** (see [11]). This means that the search text can contain search blocks, which can be referenced/reused in the replace text.

Bigex reads the search and replace expressions from two user defined text files. Both files are expected to be in the chosen **export target folder** for the IA/IGS file. If one or both exist then *Bigex* reads them and applies all valid found search and replace expressions on the exported data afterwards.

Right at beginning of the export process *Bigex* writes some status information into the console window, and if enabled into a log file. These contain information about where both search & replace text files were found and how many valid expressions were read. During further progress of the export *Bigex* prints in detail which replacements were applied.

Both text files with the replacement expressions separate in one with a fixed name and one with a fixed extension to the IA/IGS export file name. The first with the fixed name **IGS_ExpModFile.txt** resp. **IA_ExpModFile.txt** can be used for replacements which are the same over all 3D models, such as e.g. shader names.

The name for the model specific text files is built from the chosen IA/IGS export file name plus the fixed extension **_IGSExpModFile.txt** resp. **_IAExpModFile.txt**.

Example: If the chosen IGS resp. IA export file name is *MyEngine.igs*, then the resulting file name for this text file is *MyEngine_IGSExpModFile.txt* resp. *MyEngine_IAExpModFile.txt*

This type of EDM text files can be used for model specific modifications as e.g. merging of mesh objects or modification of specific material attributes.

The subfolder **Bigex_ExampleObjects** of the installation package contains four example files.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

12.2 Contents of the Bigex Export Data Modification (EDM) files

The syntax of the expressions in the base and model specific search and replace files are identical. Both can contain blank lines, comment lines, lines with search domain header or lines with search and replace expressions.

Note: Comment lines do have as **first character in the line** a # and are ignored by *Bigex* like blank lines.

12.3 Modification of object attributes

12.3.1 Search domain header

Preceding to one or a group of search and replace expressions a search domain specification is required. The search domain consists of the name of an object list and the name of the attribute to be modified.

The following section 12.3.3 IGS- and IA-object lists contains a complete list with the names of all existing object lists.

Note: Lines containing a search domain header start with a [as **very first character in the line**, followed by the object list name, a colon :, the name of the attribute of a list entry and finally a] as **last character in the line**.

All replace expressions until the next search domain header, resp. until the end of file, are applied for the search range only which is specified by the search domain header.

Example:

```
...
[Materials:ShaderName]
# Shader Name (fx)
TrDiff=TrainBasicObjectDiffuse.fx
TrSpec=TrainBasicObjectSpecular.fx

[Objects:Name]
A_FSseitFenst_Rg=1_0200_Seitenfenster_Rechts
...
```

The first two search and replace expressions are applied on the *ShaderName* attribute of all materials in the *Materials* object list.

The third search and replace expression is applied on the *Name* attribute of all objects in the *Objects* object list.

In some cases it is required to modify attributes which are more deep nested, such as e.g. the *MipLODBias* of the *RenderStage1* of a material. These more deep nested attributes can be reached by appending an underscore (*_*) followed by the name of the attribute.

Example 1:

```
...
[Materials:RenderStage1_MipLODBias]
# set all RS1 MipLODBias of all Materials
=-1.0
```

Example 2:

```
...
[IGfMaterials:PulseGlowConfig_TargColor_Green]
# set PulseGlowConfig_TargColor_Green of Material 'Mat1'
[Mat1]=0.66
...
```

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

In the first example the attribute *MipLODBias* of *RenderStage1* of all materials is set to -1.0. In the second example the value of the attribute *PulseGlowConfig TargColor Green* only for the material with name 'Mat1' is set to 0.66.

More on this in the following sections.

12.3.2 Search and replace expressions

Search and replace expressions contain of two parts. The search text at begin of a line and the replace text until the end of the same line. Both are separated by the equal sign ('=').

The search and replace expressions support regular expressions following the **Python syntax** (see [11]).

One line contains one search and replace expression in the EDM file.

Note: Lines with search and replace expressions consist of a search string, starting at the beginning of the line, ending before the equal sign =, the equal sign = itself and a replacement string, starting after the equal sign =, ending with the end of the line.

Important note: **All** characters of the search and replace text, like e.g. blanks or dots, are part of the search resp. replace text. This means all those characters are included in the search resp. included in the replacement. The only exception are the control characters for regular expressions: '(', '[', '\ and the equal sign '='.

Four different cases are differentiated for search strings.

1. **Unconditioned replacement:**

If the search string is blank, which means the equal sign = is the first character in a line, then the specified attribute of **all** objects of an object list is replaced. This is independent of the value it holds currently.

2. **Limited on objects with a certain index in a list:**

If the search string starts with **[n]** or **[n-m]**, then the **n-th** resp. the **n-th to m-th** objects of an object list are searched and only the value of their attribute is replaced.

3. **Limited on certain objects of a list:**

If the search string starts with **[text]**, then only the object with the name attribute **text** is modified. All other objects remain unchanged. If the object has no name attribute, then this search and replace expression is ignored.

4. **Limited on certain content of attributes:**

If the search string does not start with [or =, then it is searched for values of attributes, which are **exactly** matching the search string.

All these cases can be combined too.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Examples:

```

...
# Search domain: Object list Materials, Attribute ShaderName
[Materials:ShaderName]
# 1. Replaces all shader names of all materials by SName2 independently of the previous value
=SName2

# 2. Replaces the shader name of material 99 in the list by SName2 independently of the
previous value
[99]=SName2

# 3. Replaces only shader names of materials by SName2 if the shader name is SName1
SName1=SName2

# 4. Replaces the shader name of material 99 in the list by SName2, but only if it is SName1
[99]SName1=SName2

# 5. Replaces the shader names of materials 88 to 99 in the list by SName2, but only if it is
SName1
[88-99]SName1=SName2

# 6. Replaces the shader name of the material with the name MatName by SName2, but only if it
is SName1
[MatName]SName1=SName2
...
[IGfMaterials:PulseGlowConfig_TargColor_Green]
# 7. Sets for the material with the name 'Mat88' the value of the attribute
# PulseGlowConfig TargColor Green to 0.75
[Mat88]=0.75
...

```

The four EDM example files *IA* resp. *IGS_ExpModFile.txt*, *IA* resp. *IGS_specific_IGSExpModFile.txt* in the **ExampleObjects** subfolder of the installation package contain more examples for search and replace expressions.

12.3.3 IGS- and IA-object lists

Objects from the following object lists can be modified at the export into the IGS (IGf) format:

- IGfHeaders
- IGfObjects
- IGfMeshLODs
- IGfMeshes
- IGfVerts
- IGfVertexBoneBindings
- IGfTriangles
- IGfBones
- IGfMaterials
- IGfLights
- IGfSplines
- IGfGenericItems
- IGfGenericData
- IGfDataBlocks
- IGfData
- IGfFaceTagDescriptions
- IGfTextureNames

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Objects from the following object lists can be modified at the export into the IA (IAf) format:

- Doubleheaders
- Hair-trigger
- Animation
- Radio-controlled
- Locomotions

The preceding designator **IAf** resp. **IGf** can be omitted when those object list names are used in search domain headers.

Example:

```
# Search domain: Object list Materials , Attribute ShaderName
[IGfMaterials:ShaderName]
...
# is identical to:
[Materials:ShaderName]
...
```

12.3.4 Attributes of objects

All the single objects in those object lists have many different attributes. In total there are hundreds of different attributes existing. To find out the exact name spelling for an attribute to be replaced, it is best to run the export once **without** any replacement. Afterwards you can copy and paste the exact attribute name you want to modify from the log file.

Example: Replacing an object name.

Excerpt of the log file after an export **without** replacement:

```
...
=====
IGfObjects                : [ 42 ]
-----
IGfObjects [ 1 / 42 ] (390932) : 'AB_FSseitFenst_Rg'
-----
Name                    : 'AB_FSseitFenst_Rg'
NameUniqueID              : 0
MeshLODCount              : 4
MeshLODListStartOffset   : 390708 ==> IGfMeshLODs[0]
...
```

At export with the following replacement expression:

```
...
[Objects:Name]
AB_FSseitFenst_Rg=1_0500_Seitenfenster_Rechts
...
```

the object name `1_0500_Seitenfenster_Rechts` from the replacement expression is written into the IGS file in place of the Blender object name `AB_FSseitFenst_Rg`.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Excerpt of the log file after an export *with* replacement:

```

...
=====
IGfObjects          : [ 42 ]
-----
IGfObjects [ 1 / 42 ] (390932) : '1_0500_Seitenfenster_Rechts'
-----
Name                : '1_0500_Seitenfenster_Rechts'
NameUniqueID       : 0
MeshLODCount       : 4
MeshLODListStartOffset : 390708 ==> IGfMeshLODs[0]
...

```

The four EDM example files *IA* resp. *IGS_ExpModFile.txt*, *IA* resp. *IGS_specific_IGSExpModFile.txt* in the **ExampleObjects** subfolder of the installation package contain some more examples for search and replace expressions.

12.4 Action expressions

Beside the search and replace expressions there is a further group of instructions called action expressions. These are used for an automated modification of the geometry of a 3D model.

Action expressions start with the search domain header followed by the action.

Note: Lines containing a search domain header start with a [as **very first character in the line**, followed by the object list name, two colons ::, the name of the action and finally a] as **last character in the line**.

Example:

```

...
# Merge action header
[IGfObjects::Merge]
...

```

Currently there is only one action instruction existing.

12.4.1 Merging of mesh objects

With the action merge mesh objects (*IGfObjects::Merge*) one or more objects can be merged at export time into one single object. By this all vertices, faces, materials and UV-coordinates are transferred to the new merged object. As far as possible object hierarchies are kept intact resp. are senseful merged from the deleted objects.

The action expression for merging mesh objects starts with a range definition and the action instruction: *Merge*. The following lines until the next range definition, resp. the end of file, can contain one or more expressions for the merging. Those start with the name for the new object, which contains all merged objects afterwards, followed by a equal sign =. On the right hand side of the equal sign follows a comma separated list with names of the objects to be merged.

If the object on the left hand side of the equal sign is not existing, then the first object of the right hand side object name list gets an exceptional position. It is the origin object which all other objects to be merged will be appended.

If the list of objects to be merged contains names of non-existing objects, those will be ignored and all remaining and existing objects will be merged only.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Examples:

```

...
# Merge Action range definition
[IGfObjects::Merge]

# Create the new object MergedObjs from Obj1 and Obj2
MergedObjs=Obj1,Obj2

# Append Obj2 to Obj1
Obj1=Obj1,Obj2
# exactly the same
Obj1=Obj2

# Create the new object MergedObjs from Obj1, Obj2, Obj3 and Obj4
MergedObjs=Obj1,Obj2,Obj3,Obj4
# exactly the same with a regular expression
MergedObjs=Obj[1-4]
# exactly the same with a different regular expression
MergedObjs=Obj[4321]
...

```

The object specific EDM example file *IGS_specific_IGSExpModFile.txt* in the *Bigex_ExampleObjects* subfolder of the *Bigex* installation package contains some more example expressions for the *Merge* action.

Hint: It is recommended to merge objects which are on the same Level of Distance (LOD). These objects are anyway merged later at the conversion from the *IGS* into the *GeoPcDx* format (done by the Rail Works Asset Editor).

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

13 Running Bigex in command line mode

As well as its use as an IA/IGS exporter from Blender, the *Bigex* Python script can be used as an analyzer for existing IA/IGS files. This operation mode can very easily be started from a command line in a console window.

The analyser mode allows you to dump out the contents of an existing IA/IGS file into the console window or optionally into a log file. It is also possible to generate a comma separated value (csv) file which can easily be imported in a spreadsheet program like Openoffice Calc or Microsoft Excel. From there the data can be analyzed in more detail or processed further.

The behavior of the *Bigex* script in the analyzer mode can be controlled with options on the command line.

Running the *Bigex* script under Python from the command line is quite easy. You simply need to call ***bigex25.pyc*** if you are using Python 2.5, resp. ***bigex26.pyc*** if you are using Python 2.6.

Under Linux, open a new console window via the Start menu and type the command:

```
python ~/.blender/scripts/bigex25.pyc --help (for Python 2.5.x) or
python ~/.blender/scripts/bigex26.pyc --help (for Python 2.6.x)
```

Under Windows, open a new console window via the Start menu (Start => Run: "cmd" => OK) and type the command:

```
C:\Program Files\Python\python.exe <PathToBigex>\bigex25.pyc --help (for Python 2.5.x) or
C:\Program Files\Python\python.exe <PathToBigex>\bigex26.pyc --help (for Python 2.6.x)
```

It may be required to adjust the path names to the Python executable. The Blender scripts are usually in folder:

```
<programme drive letter>:\<program folder>\<Blender folder>\.blender\scripts
```

e.g.: C:\Program Files\Blender\.blender\scripts

Once successfully started, the option '--help' dumps out the following help text:

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Usage: bigex26.pyc [Options] (IGS-filename.igs|IA-filename.ia)

Options:

```

--version          show program's version number and exit
-h, --help        show this help message and exit
-l, --logging      Enables logging. Writes all information from IGS/IA
                  file into a log file.
                  usage -l [LogFileName]
                  If LogFileName is omitted, default output file name is
                  <IGS/IA-filename>_imp.log
                  Default is False.
-m, --modify       Enables IGS/IA file modifying. IGS/IA file will be
                  read in, modified and written back. Modify expressions
                  are read from specified modify file.
                  usage -m [ModFileName]
                  If ModFileName is omitted, default file name
                  '[IGS|IA]_ImpModFile.txt' is used.
                  If a modify file name is specified, it is
                  used additionally
                  Default is False.
-k, --headeronly  Dumps out header information of IGS/IA file only.
                  Default is False
-t, --timestamps  Prints timestamps with information about elapsed time.
                  Default is False
-o OBJECTLIST, --objectlist=OBJECTLIST
                  Writes all elements of OBJECTLIST list into a csv-
                  file. Default csv-file name is:
                  <IGS/IA-filename>_<objectlist>.csv
                  -o help      shows a list with all object list names
                  -o OBJECTLIST dumps out object list OBJECTLIST
                  -o all       dumps out all object lists
                  -o nonempty  dumps out all non-empty object lists
                  Default is False
-a, --addresslist Dumps out addresses of objects in IGS/IA file.
                  Default is False
-s, --summary     Dumps out a summary at the end of the export.
                  Default is False
-v, --verbose     Dumps out very detailed information.
                  Default is False
-q, --quiet       Dumps minimum information only. Writes no log file,
                  no time stamps, no summary.
                  Default is False

```

If requested by several options, new files will be created in the same folder where the source IA/IGS file resides.

Some examples for calling the script:

```
... bigex26.pyc -k -s MyIGSfile.igs
```

dumps out only the header and a short summary of the contents of *MyIGSfile.igs*. This gives a quick overview of what is in the IGS file.

```
.... bigex26.pyc -l -s -t -v MyIGSfile.igs
```

dumps complete info about the contents of the IGS file. Including all details on all objects plus summary and the elapsed time for each step.

```
... bigex26.pyc -o help MyIGSfile.igs
```

lists all object list names of the IGS file.

```
... bigex26.pyc -l -s -t -v -o all MyIGSfile.igs
```

dumps out detailed info about the contents of the IGS file. Plus one csv file is generated for each object list which holds at least one object.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

14 Sending problem reports

No piece of software is without errors. Thus, if the export script is aborting e.g. with an error message, please send an error report to the mail address listed in section 5 *Copyright, Licence, Warranty, Contact, Credits*. This helps to improve Bigex.

Before sending please consider the following:

- verify that your runtime environment is as described in the install section of this manual
- verify that the problem is repeatable at least three times
- are all objects removed from the Blender model which are **not** causing the problem?
- are all objects still in the Blender model which are causing the problem?
This may either objects which cause the exporter to abort with an error message in the console window or objects which are not properly exported or not exported at all.

To do a root cause analysis of the problem, the following information need to be available in a ZIP file:

- Output of the console window containing the whole export process which causes the problem.
- Log file, created with option *Verbose* enabled in the user *Bigex* GUI.
- Blender *.blend* file containing the object only which causes the problem.

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

15 Annex

15.1 Mapping of Blender data on the IA/IGS format

During the export into the IA/IGS format all necessary 3D model data from Blender needs to be mapped into the equivalent data field of the IA/IGS file. Because Blender has lots of sliders, input fields and buttons, it is not obvious which settings in Blender relate to which IGS file data fields.

If you set the value of a certain input field in Blender, e.g. the name of a texture image file, then you will find that value again in the belonging place in the log file i.e. in TextureNameList -> TextureName[0].

The following tables show which input fields of Blender are mapped to which data fields of the IA/IGS files. Data fields which are not listed in the tables can be set at export time using the EDM function of *Bigex*.

In the left part of the table with the header **Blender** the complete path to the input field in Blender is listed. Starting from the Blender sub-window over the panel and the tab, ending with the name of the input field/slider on the given tab.

In the middle part of the table with the header **IA/IGS file** the complete path to the data field in the IA/IGS file is listed.

The right hand side of the table contains two columns with comments and the activation status.

Note: All IA/IGS file data fields which are not listed in the tables can not be set via Blender. Those data fields are preset with senseful, fixed values. But, as all other data fields, these can be modified by using the *Bigex* EDM function at export time of course.

15.1.1 Blender Input fields for the IGS format

Overview about Blender input fields for the IGS file format:

Blender				IGS File		Comment	Status
Window	Panel	Tab	Field	IGS Object list	Attribute		Activated
Button Window	Shading Material Buttons (F5)	Links and Pipeline	MA:	Materials	Name		yes
Button Window	Shading Material Buttons (F5)	Texture	First active texture channel name	Materials	ShaderName		yes
---	---	---	---	Materials	RenderStageCount	Automatically calculated: Count of active Texture Layer - 1.	yes
Button Window	Shading(F5)	Shaders	LBias	Materials	RSx.MipLODBias	The Shading value is set for all active Render Stages.	no (Not via Python accessible)

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Blender				IGS File		Comment	Status
Window	Panel	Tab	Field	IGS Object list	Attribute		Activated
Button Window	Scene(F10)	Format	FPS	Materials	RSx.FPS	The Scene value is set for all active Render Stages.	yes
Scripts Window	<i>Bigex</i> GUI	Material	RSxArgy	Materials	RSx.UVArguments	The IGS format takes just the first 6 of the 8 arguments from the <i>Bigex</i> GUI.	yes
Button Window	Shading(F5)	Links and Pipeline	Zoffs	Materials	ZBias	Values is rounded to integer value.	yes
Scripts Window	<i>Bigex</i> GUI	Material	Viewer Facing	Materials	ViewFacing		yes
---	---	---	---	Materials	AmbientColor_Red AmbientColor_Green AmbientColor_Blue AmbientColor_Alpha	All preset with 1.0. Can be modified with EDM function.	yes
Button Window	Material Buttons	Material	Col RGB	Materials	DiffuseColor_Red DiffuseColor_Green DiffuseColor_Blue		yes
Button Window	Material Buttons	Material	Mir RGB	Materials	EmissiveColor_Red EmissiveColor_Green EmissiveColor_Blue		yes
Button Window	Material Buttons	Shaders	Emit	Materials	EmissiveStrength		yes
Button Window	Material Buttons	Material	Spe RGB	Materials	SpecularColor_Red SpecularColor_Green SpecularColor_Blue		yes
Button Window	Material Buttons	Material	A	Materials	DiffuseColor_Alpha SpecularColor_Alpha EmissiveColor_Alpha		yes
Button Window	Material Buttons	Shaders	Spec	Materials	SpecularPower	To get a larger value range for SpecularPower, the both Blender fields are multiplied.	yes
Button Window	Material Buttons	Shaders	Hard				
---	---	---	---	Headers	ObjectCount	Automatically calculated: Count of objects in object list 'IGfObjects'	yes
---	---	---	---	Headers	MeshCount	Automatically calculated: Count of objects in object list 'IGfMeshes'	yes
---	---	---	---	Headers	BoneCount	Automatically calculated: Count of objects in object list 'IGfBones'	yes

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Blender				IGS File		Comment	Status
Window	Panel	Tab	Field	IGS Object list	Attribute		Activated
---	---	---	---	Headers	MaterialCount	Automatically calculated: Count of objects in object list 'IGfMaterials'	yes
---	---	---	---	Headers	LightCount	Automatically calculated: Count of objects in object list 'IGfLights'	yes
Button Window	World Buttons	World	AmbR AmbG AmbB	Headers	AmbientColor_Red AmbientColor_Green AmbientColor_Blue		yes
---	---	---	---	Headers	AmbientColor_Alpha	Fix set to 0.0.	yes
---	---	---	---	Headers	SplineCount	Automatically calculated: Count of objects in object list 'IGfSplines'	yes
---	---	---	---	Headers	GenericItemCount	Automatically calculated: Count of objects in object list 'IGfGenericItems'	yes
3D View	---	Transform Properties	Vertex X Vertex Z Vertex Y	Verts	Point_X Point_Y Point_Z		yes
---	---	---	---	Verts	Point_Weight	Preset with 1.0. Can be modified with EDM function.	yes
3D View	---	Weight Paint Properties	Weight	Verts	Normal_X Normal_Y Normal_Z	Values are from the Blender internal data structures: Normals for this Vertice	yes
---	---	---	---	Verts	Normal_Weight	Preset with 1.0. Can be modified with EDM function.	yes
---	---	---	---	Verts	ValidUVsCount	Automatically calculated: Count of UV Layer	yes
U/V Image Editor	---	---	---	Verts	UV00, ..., UV07	Values are from the Blender internal data structures: UV-Coordinates for this Vertice	yes
3D View	Editing (F9)	Mesh Tools More	Draw Normals	Triangles	Normal_X Normal_Y Normal_Z	Values are from the Blender internal data structures: Face normals of this face	yes

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Blender				IGS File		Comment	Status
Window	Panel	Tab	Field	IGS Object list	Attribute		Activated
---	---	---	---	Triangles	Normal_Weight	Preset with 1.0. Can be modified with EDM function.	yes
---	---	---	---	Meshes	VerticeCount	Automatically calculated: Count of Vertices of this Mesh	yes
---	---	---	---	Meshes	TriangleCount	Automatically calculated: Count of Triangles of this Mesh	yes
3D View	Viewport Shading (Z, Shift-Z, Alt-Z)	---	Bounding Box	Meshes	BoundingBox_ LowerLeftPointX LowerLeftPointY LowerLeftPointZ LowerLeftPointScale UpperRightPointX UpperRightPointY UpperRightPointZ UpperRightPointScale	Automatically calculated: Maximum dimensions of the Mesh	yes
Button Window	Object (F7)	Object and Links	OB:	Objects	Name		yes
---	---	---	---	Objects	NameUniqueID	Automatically calculated: Index of the object in the object list: 'IGfObjects'	yes
---	---	---	---	Objects	MeshLODCount	Automatically calculated: Count of Mesh LODs of this object	yes
---	---	---	---	Objects	TM_ Row0_Column0 Row0_Column1 Row0_Column2 Row0_Column3 Row1_Column0 Row1_Column1 Row1_Column2 Row1_Column3 Row2_Column0 Row2_Column1 Row2_Column2 Row2_Column3 Row3_Column0 Row3_Column1 Row3_Column2 Row3_Column3	This 16 values are from the Blender internal data structures: 4x4 Transformation Matrix (TM) of the object	yes
Outliner	---	---	Object Tree	Objects	ChildrenCount	Automatically calculated: Count of child objects in the object hierarchy	yes

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

Blender				IGS File		Comment	Status
Window	Panel	Tab	Field	IGS Object list	Attribute		Activated
Button Window	Texture Buttons (F6)	Image	Image / Move File name	TextureNames	[0], ..., [N]	Automatically calculated: Material1, Texture Layer1 Material1, Texture Layer2 ... Material1, Texture LayerN Material2, Texture Layer1 ... Material2, Texture LayerN ...	yes

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

15.1.2 Blender input fields for the IA format

Overview about Blender input fields for the IGS file format:

Blender				IA File		Comment	Status
Window	Panel	Tab	Field	IA Object List	Attribute		Activated
Button Window	Scene(F10)	Anim	Sta: End:	Headers	TotalFrameCount	Automatically calculated: End frame - Start frame +1	yes
Button Window	Scene(F10)	Format	FPS:	Headers	SampleRate	Automatically calculated: To extend to value range for SampleRate the both fields FPS: and /: (Frames per second base) are multiplied	yes
Button Window	Scene(F10)	Format	/: (Frames per second base)				
---	---	---	---	Headers	AnimationNodeCount	Automatically calculated: Count of objects in the object list: 'AnimationNodes' (Is equal to count of animated mesh objects)	yes
Button Window	Objects (F7)	Object and Links	OB:	Animation Nodes	Name	Name of belonging mesh object	yes
---	---	---	---	MotionSets	MotionVector4FrameNN_X MotionVector4FrameNN_Y MotionVector4FrameNN_Z	Automatically calculated: Position of the animated mesh object in FrameNNN	yes
---	---	---	---	MotionSets	MotionQuaternionFrameNNN_a MotionQuaternionFrameNNN_b MotionQuaternionFrameNNN_c MotionQuaternionFrameNNN_d	Automatically calculated: Rotation of the animated mesh object in FrameNNN	yes

HenningBR218	Blender IA/IGS Exporter (Bigex) Manual	Version 2.1.71 2011-05-06
--------------	--	----------------------------------

15.2 References

- [1] <http://home.exetel.com.au/randomsoftware>
- [2] <http://www.blender.org/>
- [3] <http://store.steampowered.com/app/5287>
- [4] <http://www.railsimulator.com/>
- [5] <http://www.python.org> or <http://www.python.de>
- [6] <http://www.gimp.org>
- [7] <http://www.gnu.org/licenses/gpl.html>, <http://www.gnu.de/documents/gpl-3.0.de.html>
- [8] <http://rail-sim.de/forum/board38-rail-simulator-railworks/board26-editoren-tools-strecken-szenarien-lokbau/board8-lokbau-3d-design/>
- [9] <http://forums.uktrainsim.com>, <http://forums.uktrainsim.com/viewforum.php?f=312>
- [10] <http://www.railsimdownloads.com/wiki/tiki-index.php>
- [11] <http://docs.python.org/library/re.html>
- [12] <http://www.railsimulator.com>, http://www.railsimdownloads.com/files/rsdevtools/RailSim_2008-11-05-1428_88_4c_DevDocs.exe
- [13] <http://www.railsimulator.com/support.php>
http://www.railsimdownloads.com/files/rsdevtools/RailSim_2008-11-05-1428_88_4c_DevDocs.exe