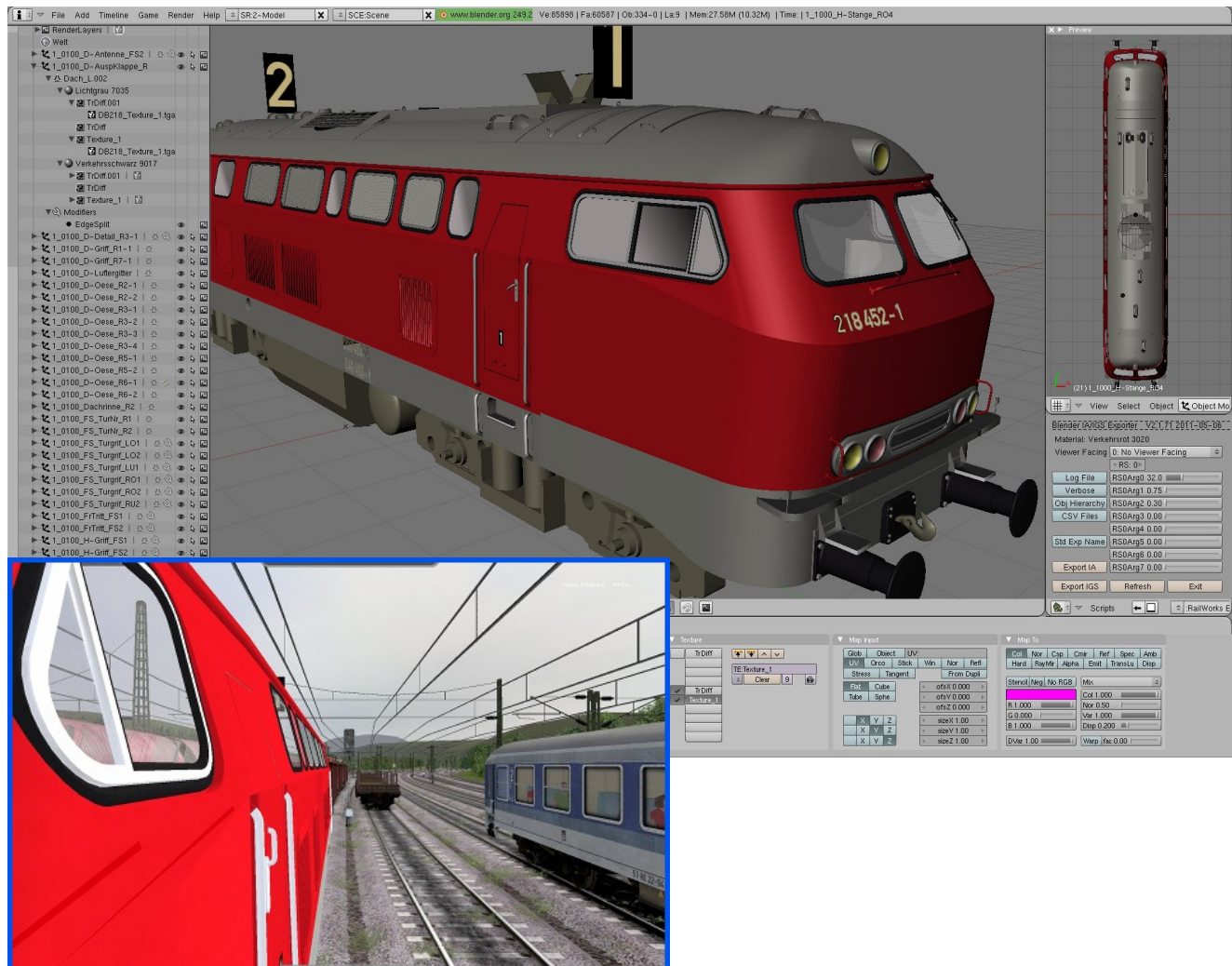


HenningBR218	Blender IA/IGS Exporter (Bigex) Handbuch	Version 2.1.71 2011-05-06
--------------	------------------------------------------------	----------------------------------



Blender IA/IGS Exporter (*Bigex*)

Handbuch

HenningBR218

Version 2.1.71 2011-05-06

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

1 Inhaltsverzeichnis

1 Inhaltsverzeichnis.....	2
2 Bildverzeichnis.....	3
3 Kurzbeschreibung.....	4
4 Inhalt und Ziel dieses Handbuchs.....	4
5 Copyright, Lizenz, Haftung, Kontakt, Dank.....	5
6 Einleitung.....	5
7 Über Bigex.....	6
7.1 Einsatzzweck.....	6
7.2 Allgemeines.....	6
7.3 Fähigkeiten im Überblick.....	6
7.4 Historie.....	7
8 Installation.....	8
8.1 Unterstützte Programmversionen.....	8
8.2 Benötigte Programme.....	8
8.3 Bigex Skripte installieren.....	8
8.4 Blender Beispieldateien.....	9
8.5 Weitere wichtige Dateien.....	10
9 Benutzung.....	11
9.1 Exportierbare Objekte.....	11
9.2 Benutzeroberfläche starten.....	11
9.3 Benutzeroberfläche.....	13
9.3.1 Eingabefelder und Knöpfe der Bigex Benutzeroberfläche.....	13
9.4 Exportvorgang.....	15
10 Erstellen von Objekten für den IA/IGS Export in Blender	17
10.1 Objekt Typen.....	17
10.2 Objekt Ausrichtung.....	17
10.3 Objekt Hierarchien.....	17
10.4 Bekannte Probleme beim Import in Rail Works.....	18
10.5 Kanten Marker.....	18
10.6 Modifizierer.....	19
10.7 Animationen.....	20
11 UV-Koordinaten- und Textur Layer.....	21
11.1 Übersicht.....	21
11.2 UV Koordinaten Layer.....	22
11.3 Textur Layer.....	23
11.4 Kompatibilität mit altem IGS Exportprogramm.....	26
11.5 Shader.....	27
12 Datenbeeinflussung beim Exportieren.....	32
12.1 Grundsätzliche Funktionsweise.....	32
12.2 Inhalt der Bigex Export Daten Modifikation (EDM) Dateien.....	33
12.3 Modifizieren von Objekteigenschaften.....	33
12.3.1 Suchbereich.....	33
12.3.2 Such- und Ersetzungsanweisungen.....	34
12.3.3 IGS- und IA-Objektlisten.....	35
12.3.4 Eigenschaften von Objekten.....	36
12.4 Aktionsanweisungen.....	37
12.4.1 Zusammenfassen von Meshobjekten.....	37
13 Bigex im Kommandozeilen Modus laufen lassen.....	39
14 Problembericht senden.....	41
15 Anhang.....	42
15.1 Abbildung der Blender Daten auf das IA/IGS Format.....	42
15.1.1 Blender Eingabefelder für das IGS Format.....	42
15.1.2 Blender Eingabefelder für das IA Format.....	47
15.2 Verweise.....	48

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

2 Bildverzeichnis

Bild 1.1: Titelbild 1.....	1
Bild 1.2: Titelbild 2.....	1
Bild 9.1: Exportierte Objekte.....	11
Bild 9.2: Benutzeroberfläche starten: Möglichkeit 1.....	12
Bild 9.3: Benutzeroberfläche starten: Möglichkeit 2.....	12
Bild 9.4: Bigex Benutzeroberfläche.....	13
Bild 10.1: Objekthierarchien.....	18
Bild 10.2: Kanten Marker.....	19
Bild 10.3: Modifizierer.....	19
Bild 10.4: Animationsparameter.....	20
Bild 11.1: UV-Textur.....	21
Bild 11.2: Texture Layer.....	22
Bild 11.3: UV-Layer.....	23
Bild 11.4: Textur Layer Stapel.....	24
Bild 11.5: Aktive Texture Layer.....	24
Bild 11.6: Render Stages.....	25
Bild 11.7: Einstellungen für den Export mit beiden Export Plugins.....	26

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

3 Kurzbeschreibung

Der Blender IA/IGS Exporter 2.1, kurz: *Bigex*, ist ein Python Plugin für das mächtige 3D-Modellierungs- und Animationsprogramm Blender. *Bigex* ermöglicht den Export in die Kuju eigenen binären Zwischenformate IGS (Modell Geometrie) und IA (Modell Animation). Diese Dateiformate werden benötigt um 3D-Modelle und soweit vorhanden ihre Animation in die Programme Rail Simulator oder Rail Works zu importieren.

Hinweis: Der Blender IA/IGS Exporter in der Version 2.1 unterstützt den Export in beide Dateiformate.

Durch die Bigex Modifikationsfunktion ist es möglich beim Export automatisiert alle Datenfelder in den IA und IGS Dateien auf beliebige Werte zu setzen. Damit hat der Benutzer die volle Kontrolle über alle Daten in den IA und IGS Dateien. Dies gilt insbesondere für die IA/IGS Datenfelder, die nicht über die Oberfläche von Blender gesetzt werden können.

4 Inhalt und Ziel dieses Handbuchs

Dieses Dokument ist das Handbuch für den Blender IA/IGS Exporter 2.1. Ziel dieses Handbuches ist es alle benötigten Informationen bereitzustellen um mit Bigex erfolgreich IA/IGS Dateien zu erstellen. Es werden in komprimierter Form alle Einstellungen und Bedienungsschritte beschrieben um sowohl einfache als auch komplexe IA/IGS Dateien zu erstellen.

Falls es für das Gesamtverständnis nötig ist werden zusätzlich noch Hintergrundinformationen gegeben. Komplexere Sachverhalte werden ausführlicher beschrieben.

Voraussetzung für das erfolgreiche Erstellen von Objekten und Animationen für den Rail Simulator oder Rail Works sind zumindest Grundkenntnisse in jedem der folgenden Bereiche:

- **Allgemein:** Shader, prinzipielle Funktionsweise von Shadern
- **Blender:** Navigieren in Blender Menüs, Erstellen einfacher 3D Objekte, Erstellen von UV Layern (Unwrapping), Erstellen und Handhaben von Blender Materialien
- **Rail Simulator / Rail Works:** Rail Simulator / Rail Works eigene Shader, genaue Shadernamen, Shader Parameter

Alle diese Punkte werden hier im Bigex Handbuch als bekannt vorausgesetzt.

Wichtiger Hinweis: Dieses Handbuch

- ist **keine** Blender Anleitung für Anfänger
- beinhaltet auch **keine** Schritt-für-Schritt Anleitung zum Erstellen von Rail Simulator / Rail Works Modellen oder Animationen
- ist auch **kein** Nachschlagewerk für Rail Simulator / Rail Works Shader Namen, Funktionsweise und Effekte

Alle diese Informationen sind aber im Internet unter den in Kapitel 15.2 *Verweise* angegebenen Links verfügbar.

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

5 Copyright, Lizenz, Haftung, Kontakt, Dank

Alle Rechte und das Copyright für die Blender IA/IGS Export Erweiterung *Bigex* liegen bei HenningBR218.

(C) HenningBR218 2009-2011

Die Blender IA/IGS Export Erweiterung *Bigex* unterliegt der GNU Public Lizenz (GPL) in der Version 3 mit den unten aufgeführten Einschränkungen für die kommerzielle Nutzung. Der komplette Text der GPL3 Lizenz ist im Internet nachzulesen (siehe [\[7\]](#)).

Die Blender IA/IGS Export Erweiterung darf nur zum Erzeugen von 3DModellen benutzt werden, die unentgeltlich weitergegeben werden (Freeware). Die Nutzung zum Erzeugen von entgeltpflichtigen 3DModellen (Payware, Shareware, etc.) bedarf der ausdrücklichen, schriftlichen Zustimmung des Autors (Kontakt: henningbr218 (at) b188 (dot) de).

Jedes mit der Blender IA/IGS Export Erweiterung *Bigex* erzeugte 3DModell ist ab einer bestimmten Anzahl von Vertices mit mehreren speziellen Markierungen versehen, die auch mit in die Rail Simulator / Rail Works 3D-Modell Dateien (GeoPcDx) übertragen werden. Die für kommerzielle Produkte lizenzierten Versionen der Export Erweiterung fügen weitere kundenspezifische Marken ein. Der Nachweis, dass ein bestimmtes 3D-Modell mit der Blender IA/IGS Export Erweiterung *Bigex* erzeugt wurde ist damit sehr leicht zu führen.

Die Benutzung von *Bigex* erfolgt auf eigene Gefahr. Jegliche Haftung des Programmautoren für direkte und indirekte Schäden an Hard- und Software, die durch *Bigex* entstehen könnten, ist explizit ausgeschlossen.

Für Fehlermeldungen, Fragen und Erweiterungswünsche kann im deutschsprachigen Forum von **Rail-Sim.de** (siehe [\[8\]](#)) und im englischsprachigen Forum von **UKTrainSim** (siehe [\[9\]](#)) ein Beitrag eingestellt werden. In Ausnahmefällen kann der Autor auch direkt via Mail kontaktiert werden: henningbr218 (at) b188 (dot) de.

Alle Rechte von in diesem Handbuch erwähnten registrierten Firmen- und Markennamen liegen bei ihrem derzeitigen Inhaber.

Ein besonderer Dank für die Mithilfe am deutschen und englischen Handbuch geht an AndiS, Jeff Douglas (JADsHome) und karma99.

6 Einleitung

Der erste Blender IA/IGS Exporter von **steampsi** (siehe [\[1\]](#)) ist inzwischen ein wenig in die Jahre gekommen. Wie allgemein bekannt ist, hat er auch ein paar Schwächen und einige Funktionen fehlen ganz. Besonders der Export von Animationen in das IA Format wird nur sehr unvollständig unterstützt. Eine weitere grosse Einschränkung ist, dass er nur unter Blender Versionen läuft, die Python 2.5 unterstützen und damit nicht mehr unter Blender Versionen grösser als 2.48.

Aus diesen Gründen war es an der Zeit ein neues Exportmodul für Blender zu schreiben. Es heisst Blender IA/IGS Exporter 2, oder kurz: *Bigex* und wurde von Grund auf neu entwickelt. Ab Bigex Version 2.1 wird auch der Export von Animationen in das IA Dateiformat unterstützt. Wie viele Blender Plugins, so ist auch *Bigex* in der Programmiersprache Python geschrieben.

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

7 Über *Bigex*

7.1 Einsatzzweck

Der Blender IA/IGS Exporter 2.1 *Bigex* exportiert animierte 3D Objekte, die mit Blender erstellt wurden, in das IGS (Intermediate Geometry Shape) oder das IA (Intermediate Animation) Dateiformat. Diese Dateiformate werden benötigt um 3D-Modelle und Animationen in die Programme Rail Simulator (siehe [\[4\]](#)) oder Rail Works (siehe [\[3\]](#)) zu importieren.

7.2 Allgemeines

Beim Export werden so viele Parameter wie möglich von Blender in das IA/IGS-Dateiformat übernommen (siehe Tabellen in Abschnitt *15.1 Abbildung der Blender Daten auf das IA/IGS Format*). Für fehlende Blender Parameter werden sinnvolle Standardwerte in die IA/IGS-Datei eingetragen.

Hinweis: *Bigex* überprüft **nicht** ob alle zu exportierenden Daten so gesetzt sind wie es die Spezifikation des Rail Simulators fordert (z.B.: Objektnamen, Shadernamen, etc.). Der Benutzer ist selber dafür verantwortlich, dass alle Parameter so gesetzt sind, dass die erzeugte IGS Datei erfolgreich in den Rail Simulator importiert werden kann.

Hinweis: Alle Aussagen in diesem Handbuch mit Bezug auf Rail Works gelten sinngemäss auch für den Rail Simulator. Beide Simulationsprogramme können die IA/IGS Dateien einlesen.

7.3 Fähigkeiten im Überblick

- Läuft unter der Linux und Windows Version von Blender (32Bit und 64Bit)
- Exportiert Animationen in das IA und 3D Objektgeometrien in das IGS Dateiformat
- Absolute, automatisierte Kontrolle über alle Werte in den IA/IGS Dateien durch eine leistungsfähige Modifikationsfunktion für exportierte Daten, einschliesslich regulärer Ausdrücke
- Unterstützt das automatische Zusammenfassen von mehreren Meshobjekten
- Exportiert wahlweise auch Objekthierarchien
- Erzeugt wahlweise auch csv Listen von allen IA/IGS Daten zum Import in ein Tabellenkalkulationsprogramm
- Erlaubt das Setzen von IA/IGS spezifischen Parametern über die Blender Oberfläche
- Unterstützt beliebige Längen für Objekt-, Shader-, Texturnamen
- Exportiert wahlweise alle sichtbaren Objekte oder nur ausgewählte Objekte
- Unterstützt mehrere UV Koordinaten Lagen (UV-Layer)
- Unterstützt mehrere Texturebenen (Texture Layer)
- Unterstützt die freie Zuordnung von UV Koordinaten Lagen zu Texturebenen
- Unterstützt die parallele Nutzung des alten Export Plugins
- Unterstützt Blender Kantenmarker (z.B.: SHARP, SEAM)
- Unterstützt Blender Modifizierer (z.B.: EDGE SPLIT)
- Enthält einen Analysemodus zum Ausgeben der Inhalte von existierenden IA/IGS Dateien als lesbaren Text oder als csv Liste zum Import in ein Tabellenkalkulationsprogramm

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

7.4 Historie

2011-05-06 Bigex Version 2.1.71 veröffentlicht

Bugfix Version:

- Problem behoben das den Start unter Blender 2.48 verhindert hat
- Problem mit absoluten Positionen von zusammengeführten Mesh Objekt Flächen behoben
- Nichtkommerzielle und kundenspezifische Modellmarkierungen verbessert
- Verschiedene kleinere Fehler behoben

2011-02-14 Bigex Version 2.1.62 veröffentlicht

Aktualisierte Version:

- IA-Format (Animation) Export
- Benutzerdefiniertes Zusammenfassen von Mesh Objekten beim Export
- Verbesserte Analyse von existierenden IGS und IA Dateien
- Unterstützung von regulären Ausdrücken für Modifikationsanweisungen
- Nichtkommerzielle und kundenspezifische Modellmarkierungen verbessert
- Verschiedene kleinere Fehler behoben

2010-04-21 Bigex Version 2.0.158 veröffentlicht

Erste Beta Version:

- IGS-Format (Modell Geometrie) Export
- Verbesserte Analyse von existierenden IGS und IA Dateien
- Automatisierte benutzerdefinierte IGS und IA Datei Modifikation beim Export
- Unterstützung für Blender 2.49 / Python 2.6

2009-02-26 Bigex Vorgänger IA/IGS Analyzer Version 1.0 veröffentlicht

Initiale Version:

- Analyse von existierenden IGS and IA Dateien
- Ausgabe des Inhalts von IGS und IA Dateien in ein lesbares Textformat

2006-06-01 Bigex Vor-Vorgänger Railsim IGS Export Preparator Version 1.0 veröffentlicht

Initiale Version:

- Vorabprüfung und Vorbereitung für den RailSim (.igs) Export

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

8 Installation

8.1 Unterstützte Programmversionen

Bigex läuft unter den folgenden Blender / Python Kombinationen:

- Blender 2.48 (32/64Bit) / Python 2.5 (32/64Bit)
- Blender 2.49 (32/64Bit) / Python 2.6 (32/64Bit)

Es können alle Unterversionen von Python 2.5 und 2.6 in der 32Bit oder 64Bit Version verwendet werden.

Bigex läuft unter Blender 2.48 und 2.49 jeweils in der 32Bit oder 64Bit Version.

Es ist geplant eine *Bigex* Version für Blender 2.5 / Python 3.0 zu erstellen, sobald eine stabile, vollwertige Blender Version 2.5 oder höher verfügbar ist.

Sowohl Blender als auch Python sind kostenlos in einer Linux und Windows Version erhältlich.

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

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

Hinweis: Der Blender IA/IGS Exporter in der Version 2.1 läuft nicht mit Blender 2.5 / Python 3.0, da bis jetzt noch keine stabile, vollwertige Blender 2.5 Version verfügbar ist.

Bigex wurde unter folgenden Betriebssystemen getestet: Linux, Windows (XP, Vista, 7).

Zusammen mit einem Bildbearbeitungsprogramm, wie z.B. dem kostenlosen GIMP (siehe [\[6\]](#)), hat man alle benötigten Werkzeuge um 3D-Objekte für die Programme Rail Simulator (siehe [\[4\]](#)) oder Rail Works (siehe [\[3\]](#)) zu erstellen. Diese Entwicklungsumgebung läuft sowohl unter Linux als auch Windows.

8.2 Benötigte Programme

Um *Bigex* benutzen zu können, müssen folgende Programme korrekt installiert sein:

- Blender, Version 2.48 oder 2.49
- Python, Version 2.5 oder 2.6
(Abhängig von der installierten Blender Version, siehe auch [8.1](#))

Hinweis: *Bigex* wurde nicht mit Blender Versionen kleiner als 2.48 getestet. Es besteht eine Chance, dass *Bigex* auch unter einer älteren Blender Version läuft, wenn diese mit Python 2.5 laufen.

8.3 Bigex Skripte installieren

Die Installation der *Bigex* Skripte ist sehr einfach. Es müssen lediglich alle Dateien, die im Unterverzeichnis **Scripts** des Installationspaketes liegen, in das Blender Skript Verzeichnis kopiert werden.

Das ist unter Linux: `<home Verzeichnis>/.blender/scripts`

also z.B.: `~/blender/scripts`

Unter Windows gibt es zwei verschiedene Möglichkeiten. Abhängig davon welche Optionen bei der Installation von Blender gewählt wurden, liegen die Blender Skripte in unterschiedlichen Verzeichnissen.

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

Falls die Option "**Blender für alle Benutzer installieren**" gewählt wurde dies:

<Windows Installationslaufwerk>:\<Dokumente und Einstellungen Verzeichnis>\<Nutzer Name>\Anwendungsdaten\Blender Foundation\Blender\.blender\scripts\

also z.B.: C:\Dokumente und Einstellungen\NutzerName\Anwendungsdaten\Blender Foundation\Blender\.blender\scripts\

Falls die Option "**Blender nur für diesen Benutzer installieren**" gewählt wurde dies:

<Programme Laufwerk>:\<Programme Verzeichnis>\<Blender Installationsverzeichnis>\.blender\scripts

also z.B.: C:\Programme\Blender\.blender\scripts

Hinweis: Falls das **.blender/scripts** Verzeichnis nicht existiert, kann es einfach neu angelegt werden.

Hinweis: Falls das **.blender/scripts** Verzeichnis im Blender Installationsordner benutzt wird (z.B.: C:\Programme\Blender\.blender\scripts), steht das Skript nur für die in diesem Verzeichnis installierte Blender Version zur Verfügung. In anderen, parallel installierten Blender Versionen ist das Skript dann nicht verfügbar.

Hinweis: Aus welchem Verzeichnis Blender die Skripte einliest kann in den Blender Benutzereinstellungen (Fenstertyp: **i User Preferences**) angegeben werden. Falls dort auf dem Reiter **File Paths** im Feld **Python Scripts** nichts eingetragen ist, werden die oben angegebenen Standardverzeichnisse nach Skripten durchsucht. Wenn etwas in den Blender Benutzereinstellungen geändert wurde, dann müssen diese Änderungen abgespeichert werden (Menü **File => Save Default Settings Ctrl-U**), ansonsten gehen alle Änderungen verloren.

Nach dem nächsten Start von Blender erscheint das **Bigex** Export Skript dann im Blender Menü (siehe 9.2 Benutzeroberfläche starten) und kann verwendet werden.

8.4 Blender Beispieldateien

Im Installationspaket sind einige Beispieldateien enthalten. Die Blender Datei **Bigex_ExampleObject_1.blend** im Unterverzeichnis **Bigex_ExampleObjects** enthält komplette Beispiele für die Benutzung von Rail Works Shadern. Die zugehörigen Texturdateien liegen im Unterverzeichnis **Textures**.

Anhand eines 3D-Modells für einige häufig benutzte Shader werden dort alle Blender Einstellungen gezeigt, die für den Export in das IGS Format benötigt werden. Ausserdem kann man dort die Anzahl der Texturlayer sehen, die der jeweilige Shader benötigt und welche Teile der Texturbilder zu welchen optischen Effekten in Rail Works gehören wie z.B. Transparenz und Glanz.

Die Beispiele sind fertig vorbereitet für den Export. Um die Beispielobjekte im Rail Railsimulator bzw. Rail Works anzuzeigen müssen die folgenden Schritte ausgeführt werden:

1. Entpacken aller Dateien aus dem **Bigex_ExampleObjects** Verzeichnis nach:
Falls **Rail Simulator** verwendet wird:
<Programme Installationslaufwerk>:\<Programmverzeichnis>\<RailSimulator Verzeichnis>\Source\HenningBR218\Addon\Bigex_ExampleObjects\

also zum Beispiel:

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

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

Falls **Rail Works** verwendet wird:

<Programme Installationslaufwerk>:\<Programmverzeichnis>\<Steam Verzeichnis>\
SteamApps\common\Railworks\Source\HenningBR218\Addon\Bigex_ExampleObjects\

also zum Beispiel:

C:\Programme\Steam\SteamApps\common\Railworks\Source\HenningBR218\Addon\Bigex_ExampleObjects\

Hinweis: Es ist nicht möglich den Namen des **Source** Ordners sowie aller darunterliegender Ordner zu ändern. Die genaue Ordnerstruktur und -namen werden von Rail Works bzw. den Voreinstellungen in *Bigex_ExampleObject_1_exp.xml* fest vorgegeben. Wenn die Ordernamen verändert werden, können die Beispielobjekte nicht mehr in Rail Works importiert werden.

2. Datei *Bigex_ExampleObject_1_exp.xml* im Asset Editor von Rail Works laden.

3. Alle Beispielobjekte nach Rail Works exportieren (Asset Editor Knopf **Export**).

Optional kann man die IGS Datei vorher auch selber exportieren. Dazu mit Blender die Datei **Bigex_ExampleObject_1.blend** laden und dann einzelne oder alle Beispielobjekte über die Bigex Benutzeroberfläche ins IGS Format exportieren (siehe Kapitel 9.2 *Benutzeroberfläche starten*).

Ein einzelnes Shader Beispielobjekt kann man exportieren indem man das betreffende Objekt mit der Maus auswählt. Ist kein Objekt ausgewählt, dann werden alle Beispielobjekte exportiert.

8.5 Weitere wichtige Dateien

Das Unterverzeichnis **Bigex_ExampleObjects** enthält noch vier weitere Dateien:

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

Sie enthalten Beispiele für die Such- und Ersetzungsausdrücke, die zur Datenbeeinflussung beim Exportieren eingesetzt werden können. Es sind z.B. für die am häufigsten benutzten Rail Works Shader Ausdrücke zum Ersetzen von abgekürzten Namen in die vollen Shadernamen enthalten, die von Rail Works erwartet werden. So zum Beispiel: *TrUpVFaceFlora=TrainUprightViewFacingFlora.fx*. Wegen der Längenbegrenzung für Namen in Blender wäre die Nutzung dieser Shader sonst nicht möglich.

Die Kommentarzeilen in diesen Dateien enthalten Beispiele für die weitere Anwendungen für die Such- und Ersetzungsausdrücke.

Wenn man also eigene Modelle exportieren möchte ist es sehr empfehlenswert die beiden Dateien **IG_ExpModFile.txt** und **IG_specific_IGExpModFile.txt** in den Ordner zu kopieren, in das die IGS-Datei exportiert werden soll. Das selbe gilt für den Export von Animationen.

Der Dateiname für die modellspezifischen Ausdrücke muss angepasst werden. Mehr dazu später in Kapitel 12 *Datenbeeinflussung beim Exportieren*.

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

9 Benutzung

9.1 Exportierbare Objekte

Grundsätzlich werden nur Objekte exportiert, die in Blender sichtbar und vom Typ **Mesh** oder **Empty** sind. Alle Objekte, die auf unsichtbar gesetzt wurden, werden nicht exportiert. Das heisst, alle Objekte, bei denen im Blender *Outliner* Fenster das Auge ausgegraut ist, werden nicht exportiert.

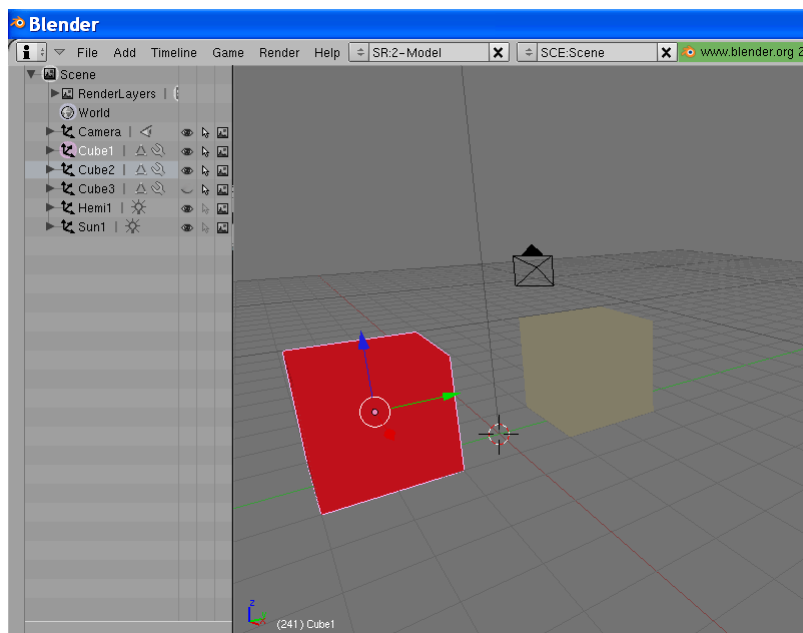


Bild 9.1: Exportierte Objekte

In diesem Beispiel wird nur das ausgewählte Mesh Objekt *Cube1* exportiert. Das nicht ausgewählte Objekt *Cube2*, das versteckte Objekt *Cube3* werden nicht exportiert.

Beleuchtungs- und Kameraobjekte werden grundsätzlich nicht exportiert da das IGS Dateiformat diese Objekte nicht kennt.

Wenn exportierbare Objekte ausgewählt sind, dann werden nur diese ausgewählten Objekte exportiert. Sind keine Objekte ausgewählt, dann werden alle sichtbaren Objekte exportiert.

Beleuchtungs- und Kameraobjekte werden grundsätzlich nicht exportiert da das IGS Dateiformat keine Datenfelder hierfür vorgesehen hat.

9.2 Benutzeroberfläche starten

Das Exportieren der Objekte erfolgt in zwei Schritten. Im ersten Schritt wird die in Bild 9.4 gezeigte Benutzeroberfläche von *Bigex* gestartet. Von dort aus erfolgt dann im zweiten Schritt über den Knopf *Export IGS* bzw. *Export IA* der eigentliche Export.

Es gibt mehrere Möglichkeiten, die Benutzeroberfläche von *Bigex* zu starten. Die einfachste ist über die Blender Menüs **File => Export => RailSim Bigex (IGS)** und **Scripts => Export => RailSim Bigex (IGS)** so wie auf den Bildern 9.2 und 9.3 zu sehen. Aber natürlich funktioniert für *Bigex*, wie auch für alle anderen Skripte in Blender, der Start mit ALT-P aus einem Textfenster, wenn man dort das *Bigex* Startskript *RS_exp_igs_V21.py* bzw. *RS_exp_ia_V21.py* geladen hat.

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

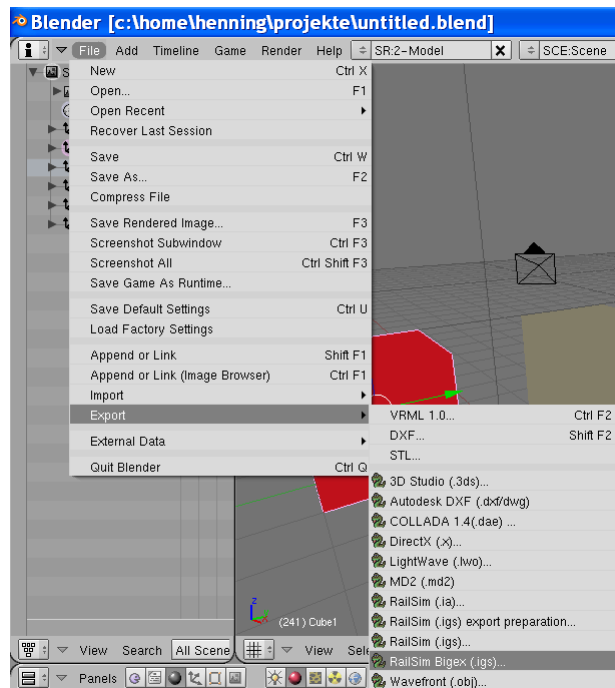


Bild 9.2: Benutzeroberfläche starten: Möglichkeit 1

Um die Benutzeroberfläche von *Bigex* über Menüs zu starten gibt es mehrere Möglichkeiten.

Die erste ist über das Menü

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

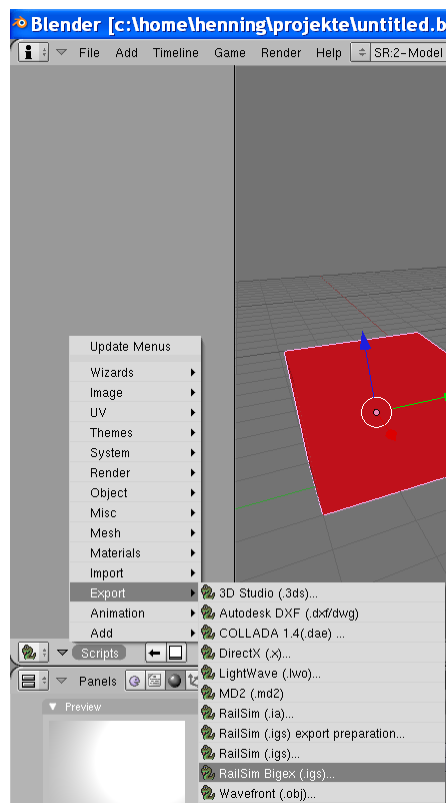


Bild 9.3: Benutzeroberfläche starten: Möglichkeit 2

Ein zweiter Weg führt über das Fenster *Scripts Window* von Blender und dort über das Menü

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

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

9.3 Benutzeroberfläche



Bild 9.4: *Bigex* Benutzeroberfläche

Die Benutzeroberfläche von *Bigex*:

Rechts sind die **Eingabefelder** für die Rail Works spezifischen Shader Parameter, die allerdings sehr selten benötigt werden.

Links sind die blauen **Optionsknöpfe** für den Export und unten die grauen **Aktionsknöpfe**.

Auf der linken Seite besitzt die *Bigex* Benutzeroberfläche Auswahlknöpfe für die IA und IGS Export Optionen: *Log File*, *Verbose*, *Obj Hierarchy*, *CSV Files* und *Std Exp Name*. Im unteren Bereich sind Aktionsknöpfe angeordnet: *Export IA*, *Export IGS*, *Refresh* und *Exit*.

Auf der rechten Seite befinden sich die Eingabefelder für sehr spezielle Parameter, die nur von einigen, wenigen Rail Works Shadern benötigt werden. Alle diese Eingabefelder werden ausschliesslich für Rail Works Material- und Shadereinstellungen benötigt. Allerdings sind diese speziellen Parameter nur für sehr wenige der Rail Works Shader definiert und werden somit entsprechend selten gebraucht.

9.3.1 Eingabefelder und Knöpfe der Bigex Benutzeroberfläche

Gleich unterhalb der Kopfzeile, im Feld **Material** wird das gerade bearbeitete Blender Material angezeigt. Es wird immer das aktuell im Panel *Shading => Material (F5)* ausgewählte Material angezeigt. Alle Eingabefelder für die Spezialparameter der Rail Works Shader beziehen sich immer nur auf das aktuell angezeigte Material.

Im Auswahlfeld **Viewer Facing** kann einer der vordefinierten Werte für diesen speziellen Parameter ausgewählt werden, den aber nur einige wenige Rail Works Shader benutzen.

Darunter im Feld **RS** kann der aktuelle Render Stage ausgewählt werden zu dem auch die folgenden acht Schiebereglerfelder **Arg** gehören. Damit können die Werte für die zusätzlichen Argumente angegeben werden, die einige wenige Rail Works Shader benutzen. Ein Rail Works Material kann bis zu acht Render Stages haben von denen jedes bis zu acht zusätzliche Argumente aufnehmen kann. Die Nummerierung sowohl der Render Stages als auch der Argumente beginnt bei Null. Die entsprechenden Bezeichner heissen also: RS0Arg0, RS0Arg1, ..., RS0Arg7, RS1Arg0, RS1Arg1, ... bis RS7Arg7.

Die allermeisten der bis jetzt bekannten Rail Works Shader benutzen keine zusätzlichen Argumente. Ein paar wenige benutzen maximal vier Argumente. Weitergehende Informationen über die Rail Works Shader sind in den Entwickler Dokumenten [\[13\]](#) und im Rail Works Wiki [\[10\]](#) zu finden.

In jedem Fall werden alle hier eingestellten Werte mit in das IGS Format exportiert, unabhängig davon ob der gewählte Rail Works Shader sie benutzt oder nicht.

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

Alle in der *Bigex* Benutzeroberfläche eingestellten Werte für alle Materialien werden mit in der .blend Datei abgespeichert und stehen somit nach dem erneuten Laden eines Modells wieder zur Verfügung.

Hinweis: Welche Werte aus Blender in welche Felder der IA/IGS Datei geschrieben werden, steht in den Tabellen in Kapitel 15.1 Abbildung der Blender Daten auf das IA/IGS Format.

In der Mitte auf der linken Seite befinden sich die blauen Knöpfe mit den Export Optionen. Mit der obersten Option **Log File** schaltet man das Mitschreiben der Ausgaben vom Konsolenfenster in eine Log Datei ein. Mit der Option **Verbose** werden noch deutlich mehr Details als im Konsolenfenster angezeigt mit in die Log Datei geschrieben. Mehr Details zu der Log-Datei sind in den folgenden Kapiteln zu finden.

Wenn die Option **Obj Hierarchy** ausgewählt wurde werden die Eltern-Kind Beziehungen (Objekt Hierarchie) von allen Objekten in Blender mit in die IGS Datei exportiert.

Die Option **CSV Files** legt fest ob zusätzlich noch Dateien mit kommaseparierten Werten (CSV) von allen Objekten und Eigenschaften angelegt werden sollen. Solche Textdateien können anschliessend sehr einfach mit einem Tabellenkalkulationsprogramm wie Openoffice Calc oder Microsoft Excel geladen und weiter analysiert oder bearbeitet werden.

Wird der Export mit eingeschalteter Option *Standard Export Dateiname (Std Exp Name)* gestartet, dann wird der Name der aktuell in Blender geöffneten .blend Datei mit der Erweiterung .ia bzw. .igs als Name für die IA bzw. IGS Exportdatei verwendet. Ein Dateiauswahl Dialog erscheint in diesem Fall nicht und der Export startet sofort.

Eine vorhandene IA bzw. IGS Datei wird ohne Nachfrage einfach überschrieben. Als kleine Sicherheit wird der im Dateiauswahl Dialog gewählte Name bzw. der Standardname automatisch noch um ein "_exp" erweitert.

Beispiel: Bei geöffneter Blender Datei *MeineLok.blend* wäre der Dateiname für die IGS-Datei mit der Modellgeometrie *MeineLok_exp.igs* bzw. *MeineLok_exp.ia* für die IA Animations Datei.

Ist die Option *Standard Export Dateiname (Std Exp Name)* beim Start des Exports ausgeschaltet, dann wird als erster Schritt des Exports ein Dateiauswahl Dialog geöffnet. In diesem kann der Dateiname der zu erstellenden IA bzw. IGS Datei angegeben werden. Dieser eingegebene bzw. ausgewählte Name wird beim nächsten Export wieder im Dateiauswahl Dialog vorgeschlagen.

Zur Erleichterung der Auswahl von vorhandenen Dateinamen im Dateiauswahl Dialog passt *Bigex* den gewählten Dateinamen automatisch den gewünschten Exportdateinamen an.

Beispiel: Die IGS Datei *MeinModell.igs* soll erzeugt werden. Dazu muss im Dateiauswahl Dialog lediglich einer der folgenden Dateien ausgewählt werden:

- *MeinModell_exp.ia*
- *MeinModell_exp.igs*
- *MeinModell_IGSexp.log*
- *MeinModell_IGSExpModFile.txt*
- *MeinModell.blend*

Die Anpassung auf den gewünschten Dateinamen *MeinModell_exp.igs* macht *Bigex* dann automatisch.

Nach der Bestätigung mit dem Knopf *Export IGS File* startet dann der Export.

Unten am Rand befinden sich die Aktionsknöpfe. Der Knopf **Exit** schliesst die *Bigex* Benutzeroberfläche. Der Knopf **Refresh** aktualisiert das aktuell angezeigte Material falls *Bigex* das nicht automatisch macht. Eine weitere Möglichkeit besteht darin, den Mauszeiger über verschiedene Unterfenster von Blender zu bewegen. Dadurch wird auch eine Aktualisierung der Werte in der *Bigex* Benutzeroberfläche ausgelöst.

Der Knopf **Export IA** bzw. **Export IGS** startet den eigentlichen Exportvorgang.

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

9.4 Exportvorgang

Nach dem Drücken des Knopfes **Export IA** bzw. **Export IGS** beginnt *Bigex* die Objekte und Einstellungen aus den internen Blender Datenstrukturen in die IA/IGS-Datenstrukturen umzusetzen. Je nach Anzahl der Objekte, deren Vertice, Polygon und Material Anzahl sowie der Ausstattung des PCs auf dem das Ganze läuft, kann das ganze einige Sekunden, Minuten oder auch eine Stunde dauern.

Beim Export arbeitet *Bigex* ausschliesslich mit Kopien der Blender Objekte. Es werden keine Objekte oder Einstellungen in Blender verändert!

Während der Export läuft, kann man den Fortschritt am Balken am oberen Bildschirmrand von Blender verfolgen. Der Fortschrittsbalken ermöglicht eine grobe Abschätzung darüber, wie lange der Export dauern wird.

Im Konsolenfenster werden während des Exports laufend wichtige Statusmeldungen sowie Warn- und Fehlermeldungen ausgegeben. Alle diese Ausgaben vom Konsolenfenster, und noch einige mehr, können optional noch in eine Log Datei geschrieben werden. Diese wird im gleichen Verzeichnis wie die IGS-Datei angelegt und heisst genauso wie die IA/IGS Datei, hat aber im Unterschied zu dieser die Endung **_IAexp.log** bzw. **_IGSexp.log**.

Beispiel: Bei geöffneter Blender Datei *MeineLok.blend* wäre der Dateiname für den IGS-Export *MeineLok_IGSexp.log* und der für den IA-Export entsprechend *MeineLok_IAexp.log*

In der Log Datei steht sehr ausführlich und in lesbarer Form, was alles in die binäre IGS-Datei geschrieben wurde.

Hinweis: Bei entsprechend grossen zu exportierenden Objekten, kann diese Log Datei sehr gross werden. Sie kann dann auch deutlich grösser als 100MByte werden!
Falls das zu Problemen führt, dann kann die Option **Verbose** ausgeschaltet werden um die Größe der Log Datei zu reduzieren.

Bigex überprüft beim Exportieren **nicht**, ob **alle** benötigten Daten angegeben, Parameter korrekt gesetzt wurden und Namen der Rail Works Spezifikation entsprechen.

Hinweis: Der Benutzer ist selber dafür verantwortlich, dass die Datenfelder in der IA/IGS-Datei, wie z.B. Objekt- und Shadernamen, Shader Einstellungen, UV-Layer, vollständig sind und der Spezifikation von Rail Works entsprechen und damit die erzeugte IA/IGS-Datei auch erfolgreich in Rail Works importiert werden kann.

Bigex erlaubt das setzen von beliebigen Werten für alle Datenfelder in den IA/IGS Dateien. Es gibt keine Einschränkungen.

Falsch bzw. gar nicht gesetzte Namen und Werte führen in der Regel dazu, dass der Asset Editor beim Importieren der IA/IGS Datei abstürzt oder der Import mit einer Fehlermeldung abgebrochen wird. In solchen Fällen müssen alle Einstellungen und Werte in Blender, die in die IA/IGS Datei exportiert werden, auf Vollständigkeit und Korrektheit geprüft werden. Das selbe gilt für die Such- und Ersetzungsanweisungen in den Modifikationsdateien.

Es werden aber Warnmeldungen ausgegeben, die auf mögliche Ursachen der Unverträglichkeit der IGS Datei für den Asset Editor hinweisen.

Falls für den IA/IGS Export wichtige Werte überhaupt nicht gesetzt wurden, wie z.B. die Farbe des Umgebungslichts, dann werden sinnvolle Werte gesetzt. Wenn wichtige Informationen wie z.B. der Dateiname für eine Textur nicht angegeben sind, wird eine Warnmeldung ausgegeben und das betroffene Objekt nur teilweise oder gar nicht exportiert.

Falls ganze Objekte oder einzelne Objektteile nicht exportiert werden, sollte man deshalb in der Log Datei nachsehen, welche Warn- und Fehlermeldungen beim Export des betreffenden Objektes ausgegeben wurden.

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

Falls keine sehr groben Fehler im Blender Objekt sind, wird am Ende des Exportvorganges die IGS-Datei geschrieben. Andernfalls wird der Export mit einer Fehlermeldung abgebrochen.

Abschliessend schreibt *Bigex* noch einen Statusbericht und einen Satz statistischer Daten in das Konsolenfenster und die Log Datei. Dort kann man dann genau nachlesen wie lange die einzelnen Exportschritte gedauert haben und wieviele Objekte, Vertices, Poligone und Materialien in die IGS-Datei geschrieben wurden.

Hinweis: Um die Ausführungszeit beim Exportieren zu verringern, empfiehlt es sich, das Konsolenfenster zu minimieren. Das Ausgeben und Rollen im sichtbaren Konsolenfenster verlängert die benötigte Zeit sehr stark. Wenn man die Ausgaben schon während des Exports ansehen möchte, kann man die Log Datei in einem externen Editor öffnen und von Zeit zu Zeit neu laden.

In einigen Fällen schlägt der Import der IA/IGS Datei in den Asset-Editor von Rail Works fehl. In anderen Fällen bricht der Export vom Asset-Editor nach Rail Works bei der Konvertierung der IGS Datei in das *.GeoPcDx* Format mit einer Fehlermeldung ab. Um auszuschliessen, dass die Ursache dafür in den Daten der IGS Datei liegt, ist es generell eine gute Idee, sich die Warnmeldungen in der *Bigex* Log Datei einmal genauer anzusehen. Wenn alle Einstellungen in Blender so abgeändert sind, dass beim Export keine Warnmeldungen mehr auftreten, dann sollte der komplette Importpfad bis nach Rail Works fehlerfrei durchlaufen.

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

10 Erstellen von Objekten für den IA/IGS Export in Blender

10.1 Objekt Typen

Grundsätzlich werden nur zwei Typen von Objekten von *Bigex* exportiert. Das sind zum einen die Blender Mesh Objekte, die die Geometriedaten eines 3D Modells enthalten und zum anderen die Blender Empty Objekte, die u.a. zum Gruppieren von Mesh Objekten dienen. Alle anderen Blender Objekte wie z.B. Splines, Kameras und Lampen, werden nicht exportiert.

10.2 Objekt Ausrichtung

Das Koordinatensystem von Blender und Rail Works ist nicht gleich. Rail Works benutzt ein linkshändiges Koordinatensystem (Positive Werte zeigen nach: X Rechts, Y Oben, Z Vorne) und Blender ein rechtshändiges (Positive Werte zeigen nach: X Rechts, Y Vorne, Z Oben). Die Umrechnung zwischen beiden Koordinatensystemen erfolgt aber automatisch durch *Bigex* beim Exportieren.

Das heisst, wenn man z.B. eine Lokomotive in Blender so baut, dass die X-Achse (roter Pfeil) in Fahrtrichtung nach rechts, die Y-Achse (grüner Pfeil) in Fahrtrichtung nach vorne und die Z-Achse nach oben zeigt, dann steht die Lokomotive auch in Rail Works richtig.

10.3 Objekt Hierarchien

Generell unterstützt das IGS Format Hierarchieebenen mit beliebiger Tiefe für Objekte. Auch *Bigex* unterstützt den Export von Objekthierarchien mit beliebiger Tiefe aus Blender in das IGS Format.

Allerdings können Objekte im IGF Format nur maximal 24 Kind-Objekte haben. Wenn ein Blender Objekt also mehr als 24 Kind-Objekte hat, dann wird für jedes weitere eine entsprechende Warnmeldung ausgegeben und das Objekt auf der obersten Hierarchieebene (root) exportiert.

Objekthierarchien werden nicht immer benötigt. Deshalb kann deren Export mit dem Optionsknopf **Obj Hierarchy** auf der Bigex Benutzeroberfläche wahlweise abgeschaltet werden. Dann werden alle Objekte auf die oberste Hierarchieebene gesetzt und es gibt somit keine Hierarchie mehr.

Falls das Konvertierungszeit *ConvertToGEO.exe*, welches vom Asset-Editor von Rail Works benutzt wird um die IGS-Dateien in das GeoPcDx Format umzuwandeln, mit der Fehlermeldung *"Instancing not supported"* abbricht, kann man versuchen die IGS Datei noch einmal ohne Hierarchien zu erzeugen.

Bild 10.1 zeigt ein Beispiel für Objekthierarchien in Blender.

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

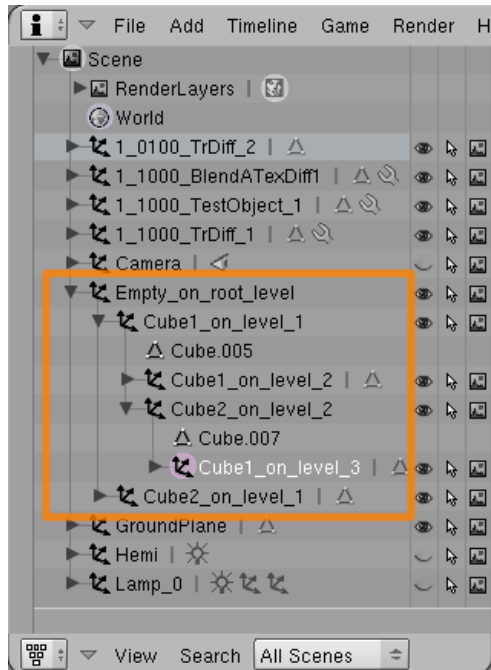


Bild 10.1: Objekthierarchien

Objekthierarchien in Blender mit einer Gesamttiefe von vier.

Das Beispiel in Bild 10.1 zeigt eine Objekthierarchie, bei der sich z.B. das Objekt *Cube1_on_level_3* auf der dritten Hierarchieebene unterhalb der Hauptebene befindet. Die komplette Ebenenstruktur wird beim Export mit ausgewählter Option **Obj Hierarchy** auch mit in die IGS Datei geschrieben.

Falls nur einige ausgewählte Objekte exportiert werden, wird versucht die Objekthierarchie soweit möglich erhalten.

10.4 Bekannte Probleme beim Import in Rail Works

Ein bekanntes Problem des Konvertierungswerkzeugs *ConvertToGEO.exe* vom Asset-Editor von Rail Works ist die Begrenzung der maximalen Anzahl der Objekte auf 256. Liegt die Anzahl darüber, bricht das Konvertierungswerkzeug mit einer entsprechenden Fehlermeldung ab. Dann bleibt nur noch die Möglichkeit, mehrere Blender Mesh Objekte zusammenzufassen um die Objektanzahl zu verringern.

Das Zusammenfassen von Meshobjekten kann entweder direkt in Blender gemacht werden, was aber nicht umkehrbar ist. Ausserdem verliert man damit die Möglichkeit einzelne Meshobjekt später noch einmal zu kopieren und z.B. gespiegelt wieder ins Modell einzufügen.

Eine elegantere Methode ist das Zusammenfassen von beliebig vielen Meshobjekten beim Export. Dies lässt das Modell in Blender unverändert. Mehr dazu in Kapitel 12.4.1 *Zusammenfassen von Meshobjekten*.

10.5 Kanten Marker

Blender bietet einige sehr nützliche Kanten Marker (Edge marker) an. Kanten, die mit einem Marker belegt sind, werden in Blender farbig hervorgehoben. Im *Editing (F9)* Menü kann man im Editiermodus auf dem Reiter *Mesh Tools More* einstellen welche Kanten farbig hervorgehoben werden. Für die weiter unten beschriebenen Marker und Modifizierer sind dies z.B. *Draw Seams* und *Draw Sharp*.

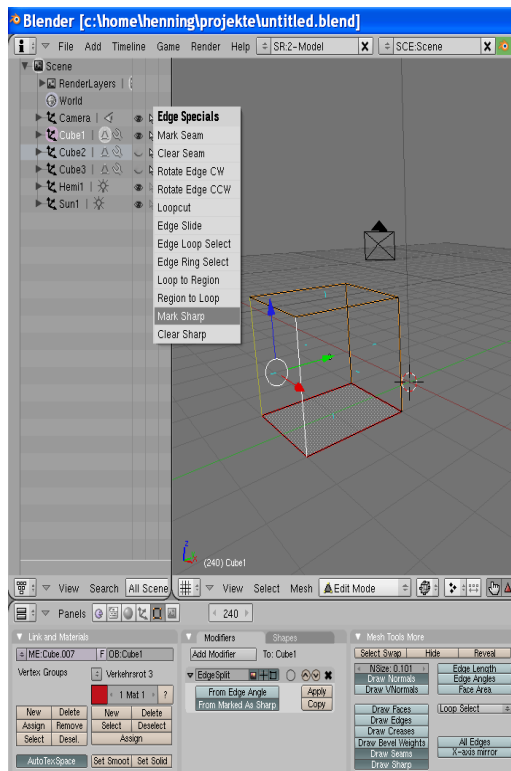


Bild 10.2: Kanten Marker

In diesem Beispiel sind im *Mesh Tools More* Menü die Flächennormale (Kleine blaue Striche, die die sichtbare Seite einer Fläche markieren), sowie die **Seam** (orange) und **Sharp** (rot) Kanten Marker auf sichtbar geschaltet.

Ausgewählte Kanten werden über das *Edge Specials* Popup Menü (CTRL-E Taste) als **Sharp** bzw. **Seam** markiert.

Als Hilfe und zum flexibleren Erstellen der UV-Koordinaten (unwrapping), bietet Blender den Kantenmarker (Edge marker) **Seam** an. Dieser Marker (orange Kanten) wird von *Bigex* unterstützt. Das heisst, ggf. werden zusätzliche Vertices für die IGS-Datei generiert, damit alle UV Koordinaten exportiert werden können.

Ein weiterer sehr nützlicher Kanten Marker den Blender anbietet, ist der Marker **Sharp** (rote Kanten). Dieser wird im Zusammenhang mit dem Modifizierer **EdgeSplit** (siehe auch 10.6) eingesetzt um scharfe Kanten an einem auf weich gesetzten (*Set Smooth*) Objekt zu erzeugen. Also z.B. bei einer geschlossenen Halbkugel. Der **EdgeSplit** Modifizierer muss dazu auf *From Marked as Sharp* stehen und *From Edge Angle* muss in diesem Fall ausgeschaltet sein.

10.6 Modifizierer

Fast alle Modifizierer werden von *Bigex* unterstützt. Modifizierer für ein Objekt werden beim Export auf eine Kopie des Objektes angewandt. Dieses kopierte Objekt wird anschliessend exportiert und danach wieder gelöscht. Das ursprüngliche Objekt bleibt also vollständig mit Modifizierern im Original erhalten.

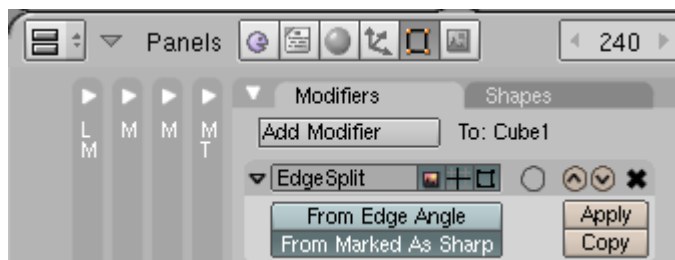


Bild 10.3: Modifizierer

Beispiel: Der sehr hilfreiche Modifizierer **EdgeSplit**.

Siehe auch Abschnitt 10.5 Kanten Marker

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

10.7 Animationen

Grundsätzlich können alle in Blender erstellten Animationen von Bigex in das IA Format exportiert werden. Dabei ist es egal ob zur Erstellung der Animation Armatures/Bones, Constraints oder IPO Keys verwendet wurden.

Für welche Objekte eine Animation exportiert wird folgt der Auswahl beim IGS Export. Wenn Objekte ausgewählt sind, dann wird nur die Animation für diese Objekte exportiert. Falls keine Objekte ausgewählt sind werden die Animationen für alle Objekte exportiert. Wenn statische Objekte exportiert werden, also Objekte die sich nicht bewegen, dann werden diese auch als unbewegte Objekte mit in die IA-Datei exportiert.

Beim Export von Animationen werden alle Frames exportiert, die im *Scene (F10)* Panel von Blender auf dem Reiter *Anim* eingestellt sind. Dabei ist der Frame mit der Nummer die im **Sta** Eingabefeld gesetzt ist der erste exportierte Frame und der im **End** Eingabefeld der letzte.

Die Framerate für den IA Export ergibt sich aus der **Multiplikation** zweier Werte vom Reiter *Format*. Der erste Wert ist vom Eingabefeld **FPS** und der zweite von *Frames per second base*, gleich rechts neben dem **FPS** Eingabefeld. Eine Multiplikation der beiden Werte ist nötig, da der Zahlenbereich für das **FPS** Eingabefeld von Blender leider auf 120fps begrenzt ist, für Rail Works aber durchaus höhere Werte angegeben werden können.

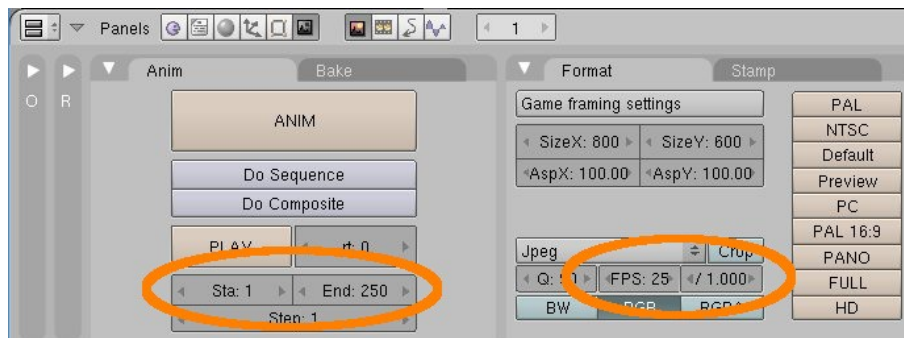


Bild 10.4:
Animationsparameter
Zu exportierender Start-
und Endframe und die
Framerate.

Der Export in das IA Dateiformat wird durch eine Klick auf den Knopf **Export IA** auf der Bigex GUI gestartet.

Wenn die Option *Standard Export Dateiname (Std Expo Name)* ausgeschaltet ist, wird als erster Schritt des Exports ein Dateiauswahl Dialog geöffnet, in dem der Dateiname der zu erstellenden IA Datei abgefragt wird. Andernfalls wird standardmässig der Name der aktuell in Blender geöffneten *.blend* Datei mit der Erweiterung *.ia* als Name für die IA Exportdatei verwendet und der Dateiauswahl Dialog erscheint nicht.

Falls sich die gemeinsame oder modellspezifische Textdatei mit den Such- und Ersetzungsanweisungen im gewählten Zielverzeichnis befindet, dann werden die Daten für die IA Datei beim Export entsprechend modifiziert. Mehr zum Thema später in Kapitel 12 Datenbeeinflussung beim Exportieren.

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

11 UV-Koordinaten- und Textur Layer

11.1 Übersicht

Um eine Textur für ein 3D-Modell zu erstellen, braucht man **UV-Koordinaten**, ein **Texturbild** und einen **Shadernamen**.

In dieser Übersicht werden diese Begriffe kurz erklärt. Die folgenden drei Kapitel gehen dann tiefer auf diese Begriffe und ihre Anwendung ein.

Die **UV-Koordinaten** beschreiben für jeden Vertex welche Position des Texturbildes auf dem Vertex liegt. Da ein Texturbild nur zweidimensional ist, benötigt man hierfür nur ein Koordinatenpaar pro Vertex. Also die X und Y Position auf dem Texturbild, die auf dem zugehörigen Vertex liegen soll. Damit es nicht zu Verwechslungen mit den X und Y Koordinaten der Vertex Position im 3D Modell kommt, werden die Koordinaten auf dem Texturbild mit U und V bezeichnet.

Eine Liste mit den U und V Koordinatenpaaren für alle Vertices eines Mesh Objekts wird **UV-Koordinaten Layer**, oder kurz: UV-Layer, genannt. In Blender kann man jedem UV-Layer einen eigenen Namen zuweisen, um sie später besser unterscheiden zu können. Es können mehrere UV-Layer mit jeweils eigenem Namen für ein Mesh Objekt angelegt werden.

In Blender wird für diese UV-Layer allerdings die Bezeichnung „UV Texture“ benutzt, was aber auf gar keinen Fall mit den weiter unten beschriebenen „Texture Layern“ verwechselt werden darf. Hier im Dokument bleiben wir deshalb bei der Bezeichnung UV-Layer.

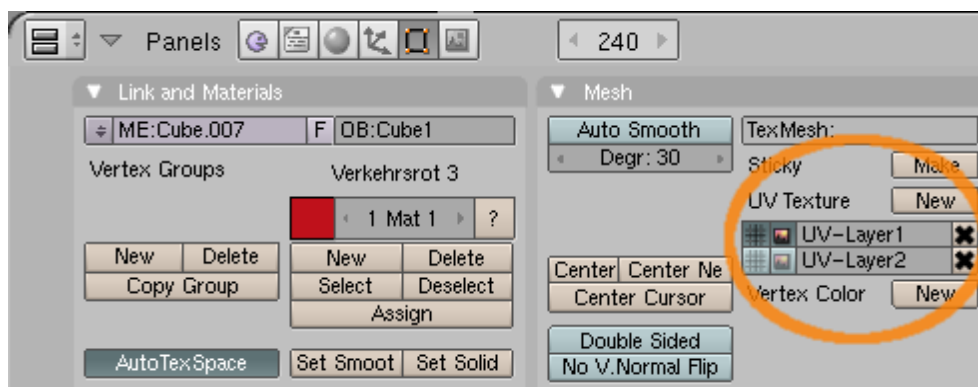


Bild 11.1: UV-Textur

Zwei UV-Layer mit jeweils einem Satz UV Koordinaten für ein Mesh Objekt.

Ein **Texturbild** wird in Blender über **Textur Layer** festgelegt. Je Textur Layer kann ein Texturbild angegeben werden. Weiterhin kann festgelegt werden, welcher der UV-Layer für dieses Texturbild benutzt wird um es auf die Vertices eines Mesh Objekts abzubilden. Auch hier ist es möglich mehrere Textur Layer für ein Material in Blender anzulegen und mit einem jeweils frei wählbaren Namen zu bezeichnen. Ein Textur Layer kann sowohl innerhalb eines Blender Materials als auch für weitere Materialien beliebig oft wiederverwendet werden.

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

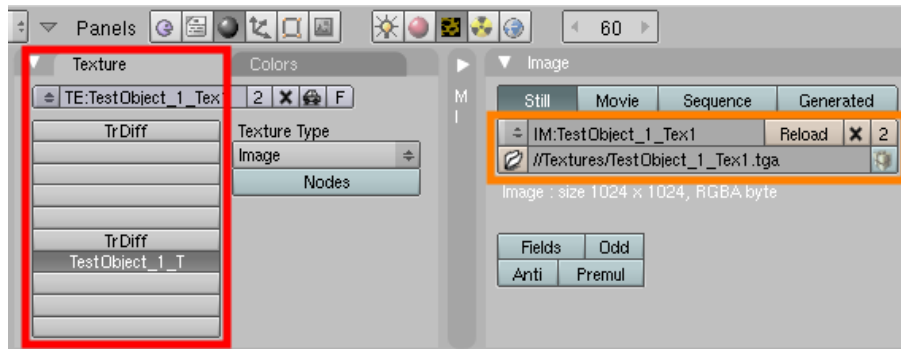


Bild 11.2: Texture Layer

Der *Textur Layer* Stapel von Blender im *Texture Menü* (rote Umrandung) mit drei aktiven Layern. Dem ausgewählten siebten Layer *TestObject_1_T* ist das Texturbild (Image) *TestObject_1_Tex1.tga* zugeordnet (orange Umrandung).

Wichtiger Hinweis: Die im *UV/Image Editor* Fenster beim Erstellen der UV Koordinaten ausgewählten Bilder sind vollkommen unabhängig von den Texturbildern, die in den Textur Layern angegeben werden müssen! Die *UV/Image Editor* gewählten Bilder werden nur im Blender 3D Fenster benutzt. Für den Export in das IGS Format und damit die spätere Nutzung in Rail Works, sowie für gerenderte Bilder in Blender, werden ausschliesslich die in den *Textur Layern* angegebenen Texturbilder benutzt!

Damit Rail Works die Flächen eines 3D-Modells darstellen kann, muss ein **Shadername** angegeben werden. Die verfügbaren Rail Works Shader sowie deren Namen und Eigenschaften werden von Rail Works also fest vorgegeben. Je nach gewähltem Shader benötigt dieser eine, zwei oder drei Textur Layer mit jeweils einem zugewiesenen Texturbild. Die Rail Works Shadernamen werden Blender über den Namen eines Textur Layer mitgeteilt (siehe 11.3 Textur Layer).

11.2 UV Koordinaten Layer

UV Layer sind Listen, die die zweidimensionalen Koordinaten von Punkten auf einem Texturbild für jeden Vertex eines 3D Objektes enthalten. Die UV Koordinaten werden später vom Shader in Rail Works benötigt, um die Texturbilder punktgenau auf die 3D Objekte abzubilden.

Normalerweise ist ein UV Layer ausreichend. In seltenen besonderen Fällen kann es aber sinnvoll sein, einen zweiten oder dritten UV Layer anzulegen. Etwa wenn man ein zweites, eigenes Texturbild für z.B. das Normal Mapping oder eine Glanztextur verwendet, welches aber anders auf den Flächen liegen soll oder nur für einige Flächen benötigt wird.

UV-Koordinaten werden in Blender neu erstellt, indem man im Edit Modus die Flächen auswählt, für deren Vertices man UV-Koordinaten haben möchte. Dann benutzt man die „auswickeln“ Methode (unwrap) von Blender. Diese erreicht man im *Edit Mode* mit der Taste **U** im Menü *UV Calculation*. Das Menü zeigt gleich eine ganze Liste von verschiedenen Verfahren zum unwrappen. Welche davon das beste Ergebnis bringt muss man ausprobieren.

Nach diesem Schritt sind die UV-Koordinaten der ausgewählten Vertices zum aktiven UV-Layer hinzugefügt worden. Wenn vorher noch kein UV-Layer existierte, wurde ein neuer Layer angelegt.

Den neuen UV-Layer findet man bei ausgewähltem Mesh-Objekt unter *Editing (F9)* => Karteikarte *Mesh* direkt unterhalb von *UV Texture*.

Weitere UV-Layer erstellt man mit dem Knopf *New*. Zu beachten ist aber, dass neue UV-Koordinaten immer nur zum **aktiven** UV-(Koordinaten) Layer hinzugefügt werden.

Die Namen der UV-Layer können beliebig gesetzt werden. Hier im Beispiel wurden sie auf **UV-Layer1**, **UV-Layer2** und **UV-Layer3** gesetzt, wobei **UV-Layer2** aktiviert ist.

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

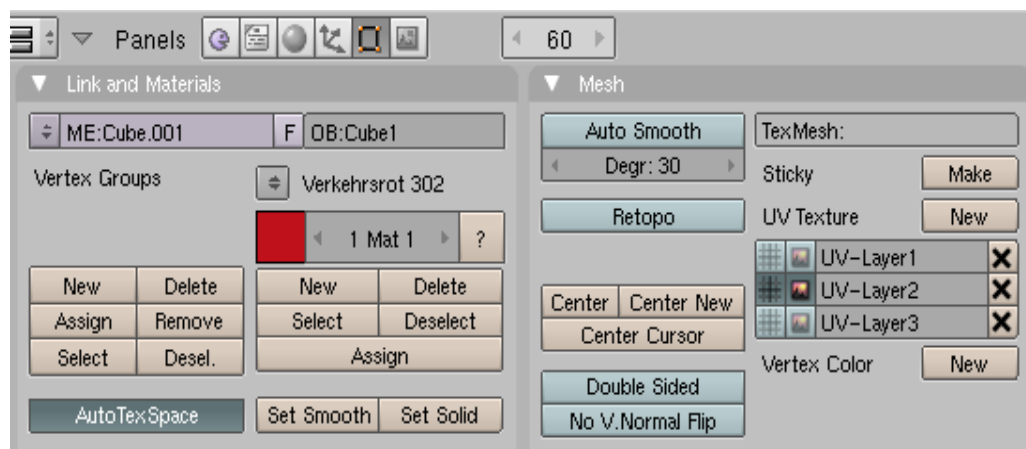


Bild 11.3: UV-Layer

UV-Layer, in Blender zu finden im *Mesh* Menü unter dem Begriff *UV Texture*.

11.3 Textur Layer

Textur Layer werden in Blender benutzt um die Parameter für die einzelnen Render Stufen für Rail Works Shader festzulegen. Texturlayer gehören zu Blender Materialien.

Jeder Fläche von jedem Mesh Objekt in Blender muss ein Material zugeordnet sein, um sie in das IGS Format exportieren zu können. Dieses Material beschreibt im Detail, wie ein 3D-Objekt später in Rail Works dargestellt wird. Über das Material werden also die visuellen Eigenschaften festgelegt.

In Blender gibt es viele Parameter mit denen diese Eigenschaften festgelegt werden können. Von der sehr grossen Anzahl von Parametern kennt das IGS Format aber nur einige wenige.

Eine der wichtigsten Eigenschaften von Materialien in Blender sind die Textur Layer. Sie legen den von Rail Works für dieses Material benutzten Shader fest und referenzieren die Texturbilder für die Shader Durchgänge.

Pro Textur Layer kann eine Datei angegeben werden, die das Bild für diesen Textur Layer enthält. Die Bilddateien können vom Format **tga** oder **png** sein, weil diese beiden Formate auch einen Alpha-Kanal enthalten können. Es sollte aber bevorzugt das tga Format verwendet werden.

Die Textur Layer Verwaltung findet man in Blender Material Menü:

(*Shading (F5)* => *Material Buttons* => Tab: *Texture*). Dort kann für jedes Material ein ganzer Stapel von Textur Layern definiert werden. Um einen Textur Layer zu definieren, kann man entweder einen existierenden Layer auswählen oder einen neuen erstellen. Ein neuer Textur Layer wird mit dem Knopf *Add New* angelegt. Anschliessend setzt man den Typ im Feld *Texture Type* auf *Image* und wählt mit der Dateiauswahl auf dem Tab *Image* die Texturdatei für diesen Textur Layer wie im Bild 11.2 gezeigt aus.

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

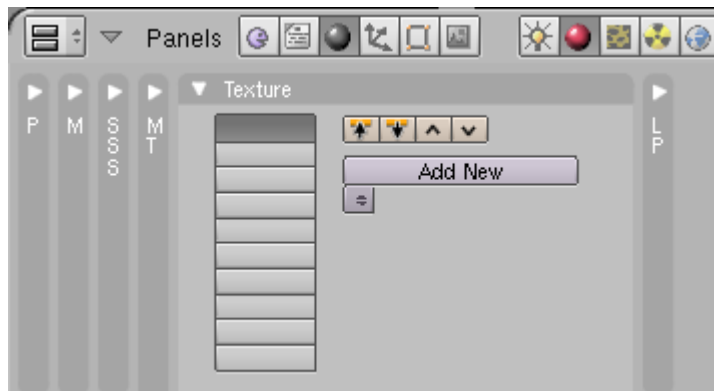


Bild 11.4: Textur Layer Stapel

Der leere *Texture Layer* Stapel in Blender, diesmal im *Material* Menü.

Wie schon erwähnt benötigt Rail Works seine eigenen, spezifischen Shadernamen um 3D Modelle korrekt darstellen zu können. Um diesen Shadernamen in Blender zu setzen wird ein bestimmter Blender Textur Layer ausschliesslich dazu benutzt, diesen Shadernamen zu setzen.

Weitere Informationen zu Rail Works Shadern sind in Kapitel 11.5 Shader beschrieben.

Wichtiger Hinweis: Der Shadernamen für ein Material, der zum Export in das IGS Format benutzt wird, wird in Blender über den Namen des ersten aktiven Textur Layers bestimmt.

Wichtiger Hinweis: Einige Shadernamen werden wegen der Begrenzung der Namenslänge in Blender nur in Kurzform angegeben (siehe 11.5 Shader) und später beim Exportieren durch die vollen Rail Works Shadernamen ersetzt (siehe 12 Datenbeeinflussung beim Exportieren).

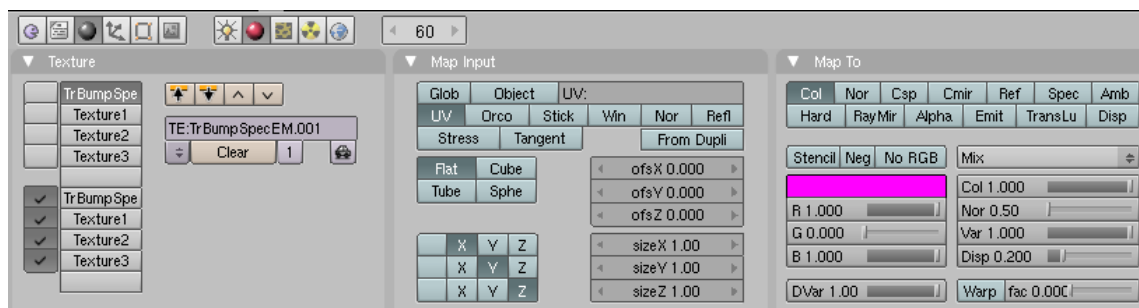


Bild 11.5: Aktive Textur Layer

Im Beispiel in Bild 11.5 ist der sechste Texture Layer der Liste der erste aktive. Er legt den Shadernamen *TrBumpSpecEM* fest. Über die folgenden drei aktiven Texture Layer werden die für diesen Rail Works Shader benötigten drei Texturdateien festgelegt.

Von diesem ersten, aktiven Textur Layer wird **ausschliesslich** der Name benutzt. Ansonsten werden hiervon **keine** weiteren Parameter in das IGS Format exportiert. Dadurch sind die Einträge für die Shadernamen sauber von denen für die Texturdefinitionen getrennt. Deshalb braucht man auch nur einen Textur Layer pro Rail Works Shadernamen anzulegen. Diesen kann man dann für andere Materialien beliebig oft wieder verwenden.

Die eigentlichen Textur Layer für den gewählten Shader werden dann in einen der Einträge nach dem Shadernamen eingetragen. Abhängig davon wieviele Textur Layer der gewählte Shader erwartet, muss die entsprechende Anzahl in die Textur Layer Liste eingetragen und aktiviert werden. Da die Namen dieser Textur Layer nicht für den Export verwendet werden, sondern nur deren Texturbild Referenz und einige der Einstellungen, können diese Textur Layer ebenfalls beliebig oft für andere Materialien wiederverwendet werden.

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

Wichtiger Hinweis: Leere Textur Layer sowie inaktive Textur Layer (ohne Häkchen) werden beim Export ignoriert.

Das bedeutet, dass der nächste **aktive** Textur Layer nach dem Layer mit dem Shader Namen, als erste Texturlage für den Shader exportiert wird. Der folgende aktive für die zweite Lage, usw. Es werden alle folgenden aktiven Textur Layer exportiert, unabhängig davon wie viele Layer der Shader wirklich erwartet und nutzt.

Hinweis: *Textur Layer* werden in Rail Works **Render Stages** (Render Stufen) genannt. Beide Begriffe bezeichnen die selbe Eigenschaft von Shadern. Alle Blender *Texture Layer*, bis auf den ersten für den Shadername, werden eins zu eins in die IGS Datei exportiert und damit auf die Rail Works **Render Stages** abgebildet.

Von den Textur Layern für den gewählten Shader werden die folgenden Parameter zum Export in das IGS Format genutzt:

Karteikarte **Map Input**, Auswahlknopf **UV** Muss immer ausgewählt sein.

Karteikarte **Map Input**, Eingabefeld **UV**: Wenn mehrere UV Layer angelegt wurden, dann kann hier festgelegt werden, welcher UV Layer für diesen Textur Layer benutzt werden soll. Falls hier nichts angegeben wird, wird der **aktive** UV Layer verwendet. Normalerweise wird nur ein UV Layer angelegt und dieses Feld bleibt leer.

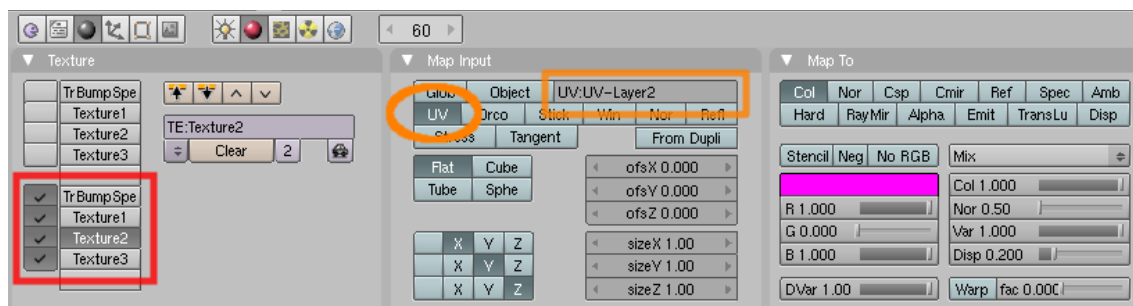


Bild
11.6:
Render
Stages

Im Beispiel in Bild 11.6 sind vier Textur Layer aktiviert (rote Umrandung). Für den zweiten Texturdurchgang (Render Stage) vom Shader *TrBumpSpecEM* ist festgelegt, dass die UV Koordinaten vom UV Layer mit dem Namen *UV-Layer2* benutzt werden sollen.

Einige weitere Parameter von Textur Layern werden auch mit in die IGS Datei exportiert. Eine detaillierte Auflistung zeigt die Tabelle in Kapitel 15.1.1 Blender Eingabefelder für das IGS Format.

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

11.4 Kompatibilität mit altem IGS Exportprogramm

Falls man sein 3D Modell wahlweise auch mit dem alten IA/IGS Exportprogramm exportieren möchte, gibt es eine einfache Lösung. Dies ist zwar ab der aktuellen *Bigex* Version 2.1 nicht mehr nötig, da diese jetzt auch den Export von Animationen in das IA Dateiformat unterstützt. Es funktioniert aber ohne Probleme.

Die Lösung besteht darin, zwei getrennte Textur Layer Gruppen anzulegen. Eine nicht aktivierte für das alte IGS Export Plugin und eine aktivierte für *Bigex*. Allerdings erwarten beide Export Skripte den Shadernamen als Namen eines bestimmten Textur Layers. In Blender kann es aber keine zwei Textur Layer mit gleichem Namen geben, die unterschiedliche Einstellungen verwenden.

Dieses Problem kann aber leicht umgangen werden. Das alte IGS Export Plugin von Blender (siehe [1]) erwartet den Shadernamen im **ersten** Eintrag der Textur Layer Liste. Es akzeptiert die Kurzform der Shadernamen von Rail Works, aber auch eine Erweiterung des Shadernamens in der Form „**.001**“, **.002**, ...“. Durch die Erweiterung des Shadernamens / Textur Layer Namens für den alten IA/IGS Exporter lassen sich somit die beiden Textur Layer für die Shadernamen sauber trennen.

Ab dem **zweiten** Eintrag der Textur Layer Liste erwartet der alte IA/IGS Exporter, abhängig vom gewählten Shader, ggf. noch weitere Textur Layer. Für alle weiteren, nicht benötigten Textur Layer werden zwar Warnungen ausgegeben, diese aber ansonsten ignoriert. Dem alten Exporter ist es im Unterschied zu *Bigex* aber egal ob diese Textur Layer aktiv sind oder nicht.

Die rot umrandete Textur Layer Gruppe in Bild 11.7 entspricht also dem was der alte Exporter erwartet.

Die aktivierte Gruppe darunter entspricht dem was *Bigex* an Textur Layern erwartet.

Wie eingangs erwähnt, läuft der alte Exporter nur unter Blender 2.48 und Python 2.5. Wenn man diese Kombination laufen hat, dann kann man wahlweise mit beiden Exportern exportieren **ohne** irgendwelche Veränderungen vornehmen zu müssen.

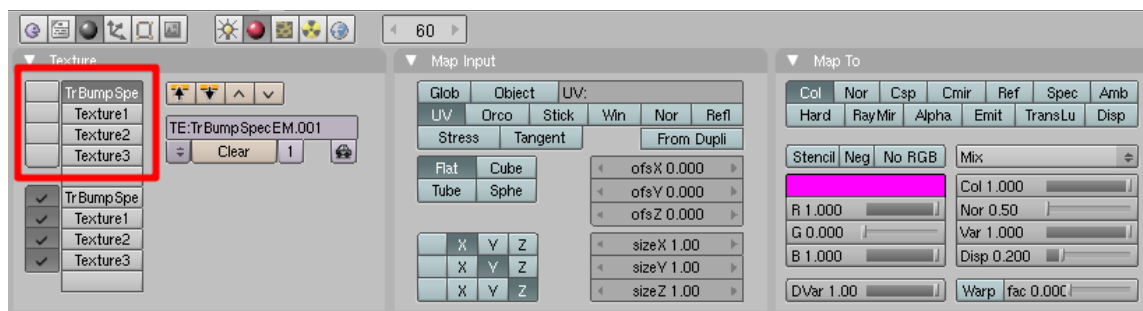


Bild 11.7: Einstellungen für den Export mit beiden Export Plugins

Hinweis: Alle Textur Layer, ausser denen für die beiden Shadernamen, können in beiden Gruppen gleichzeitig, und auch mehrfach benutzt werden. Der alte Exporter benutzt von diesen ausschliesslich den Texturdateinamen. *Bigex* dagegen benutzt noch weitere Einstellungen (siehe Kapitel 15.1.1 Blender Eingabefelder für das IGS Format)

Beim alten Export ist noch zu beachten, dass die Erweiterung der Kurzform der Shadernamen in die lange Form fest eingebaut ist. Es müssen also genau die Shaderkürzel aus der Dokumentation verwendet werden. Da die Shadernamen fest einprogrammiert sind, kennt der alte Exporter die neuen Rail Works Shader nicht und kann mit den geänderten Shadern nicht mehr korrekt umgehen. Diese Shader können mit dem alten Exporter also nicht mehr verwendet werden.

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

11.5 Shader

Die folgende Tabelle listet alle derzeit bekannten Rail Works Shader auf.

Hinweis: Da sowohl die Anzahl der Shader, als auch die Parameter, jederzeit vom Rail Works Entwickler Team geändert und automatisch via steam installiert werden können, ist die Tabelle unter Umständen nicht vollständig und aktuell.

Die Tabelle enthält in der ersten Spalte die offiziellen, voll ausgeschriebenen Rail Works Shadernamen. Die zweite Spalte enthält, soweit in der Datei mit den Modifikationsausdrücken definiert, die Kurzform des Shadernamens für die Nutzung in Blender.

In den weiteren Spalten stehen, soweit bekannt, Kurzbeschreibung, Anzahl der Textur Layer und ein kurze Beschreibung der einzelnen Texturen / Textur Layer / Render Stages.

Weitere und aktuellere Informationen über die Rail Works Shader sind in der *Rail Works Developer Package* Dokumentation (siehe [\[12\]](#)) und im RS WIKI (siehe [\[10\]](#)) zu finden.

Die Shader teilen sich in zwei Gruppen auf: **Nicht-fx** Shader und **fx** Shader.

Die Texturbilder unterscheiden sich in der Farbtiefe:

- RGB-Bilder mit 24Bit pro Pixel (8Bit Rot, 8Bit Grün, 8Bit Blau)
- RGBA-Bilder mit 24Bit RGB-Bild und 8Bit Alphakanal (Graustufen) mit insgesamt 32Bit pro Pixel.

Non-fx Shader					
Shader Name (Non-fx)	Kurzform	Beschreibung	Textur 1	Textur 2	Textur 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: Farbe 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: Farbe A: Transp.		
AddDiffuse		No texture, with diffuse colour, using additive alpha			
AddTex	AddTex	Texture mapped, no lighting applied, using additive alpha	RGB: Farbe		
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) Handbuch	Version 2.1.71 2011-05-06
--------------	------------------------------------------------	----------------------------------

Non-fx Shader					
Shader Name (Non-fx)	Kurzform	Beschreibung	Textur 1	Textur 2	Textur 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 lightmaps if used			
DualTexDiffAddWithoutLightIntens		Add second pass to first pass, brightness of second pass not affected by lightmaps 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) Handbuch	Version 2.1.71 2011-05-06
--------------	------------------------------------------------	----------------------------------

Non-fx Shader					
Shader Name (Non-fx)	Kurzform	Beschreibung	Textur 1	Textur 2	Textur 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		Bumpmap 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: Farbe		
TexDiff	TexDiff	Texture mapped with single texture, diffuse colour applied	RGB: Farbe		
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 lightmaps if used			
TripleGlossMapWithoutLightIntens		Triple texture, 2nd pass alpha channel gloss map, 3rd pass drawn additively not affected by lightmaps 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) Handbuch	Version 2.1.71 2011-05-06
--------------	------------------------------------------------	----------------------------------

fx-Shader					
Shader Name (fx)	Kurzform	Beschreibung	Textur 1	Textur 2	Textur 3
TrainEnv.fx	TrEnv		RGB: Farbe	RGB: Dummy	keine
LoftTexDiff.fx	LoftTexDiff		RGB: Farbe		
LoftTexDiffTrans.fx	LoftTexDiffTr		RGB: Farbe 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: Farbe A: Transp.		
SkinGloss.fx		Textured, normal mapped, specular with gloss map, and skinned.	RGB: Farbe	RGB: Normal Map	RGB: Gloss Map
SkinNormal.fx		Textured, normal mapped, specular and skinned.	RGB: Farbe	RGB: Normal Map	
SkinSpecular.fx		Textured, specular and skinned.	RGB: Farbe		
StencilShadow.fx	Shadow	Stencil shadow objects, material must begin with shadow_ to be detected	RGB: Farbe		
TrainBasicObjectDiffuse .fx	TrDiff	Single texture, dynamic lighting.	RGB: Farbe		
TrainBasicObjectSpecular.fx	TrSpec	Texture, colour modulated specular.	RGB: Farbe A: Transp.	RGB: Spec color map	
TrainBumpEnv.fx		Textured, normal mapped, environment mapped.	RGB: Farbe	RGB Normal Map	RGB: Dummy (Cubic Env)
TrainBumpEnvMask.fx		Textured, normal mapped, masked environment map.	RGB: Farbe A: Env Mask	RGB: Normal map	RGB: Dummy (Cubic Env)
TrainBumpSpec.fx	TrBumpSpec	Textured, normal mapped, specular.	RGB: Farbe A: Transp.		
TrainBumpSpecEnv.fx	TrBumpSpecEM	Textured, normal mapped, environment map and specular.	RGB: Farbe	RGB Normal Map	RGB: Dummy (Cubic Env)
TrainBumpSpecEnvMask.fx		Textured, normal mapped, masked environment map and specular.	RGB: Farbe A: Env & Spec Mask	RGB: Normal map	RGB: Dummy (Cubic Env)
TrainBumpSpecMask.fx		Textured, normal mapped, masked specular.	RGB: Farbe A: Env Mask	RGB: Normal map	
TrainFlora.fx	TrFlora	Ambient lighting, single texture.	RGB: Farbe		
TrainGlass.fx		Screen space refractive glass with normal map and diffuse.	RGB: Farbe	RGB: Normal map	Back buffer copy

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

fx-Shader					
Shader Name (fx)	Kurzform	Beschreibung	Textur 1	Textur 2	Textur 3
TrainLightMapWithDiffuse.fx	TrLightMap	Diffuse tex, lightmap, dynamic lighting.	RGB: Farbe	RGB Lightmap	
TrainSkyDome.fx	Sky	Skydome	RGB: Farbe	RGB: Dummy (Cubic Env)	
TrainSpecEnv.fx		Textured, vertex environment mapped with specular.	RGB: Farbe	RGB: Dummy (Cubic Env)	
TrainSpecEnvMask.fx	TrSpecEM	Textured, masked vertex environment mapped with specular.	RGB: Farbe A: Env & Spec Mask	RGB: Dummy (Cubic Env)	
TrainUprightViewFacingFlora.fx	TrUpVFaceFlora	Single texture, globally lit, upright view facing	RGB: Farbe A: Transp.		
TrainVertexLit.fx		Diffuse tex, vertex lighting only.	RGB: Farbe		
TrainVertexLitWithDiffuse.fx		Diffuse tex, vertex lighting, dynamic lighting.	RGB: Farbe		
TrainViewFacingFlora.fx	TrVFaceFlora	Single texture, globally lit, view facing	RGB: Farbe A: Transp.		
WaterCubeMap.fx	Water	Splish	RGB: Farbe 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: Farbe	RGB: Dummy (Cubic Env)	
TrainBumpEnvMask.fx		Textured, masked vertex environment map.	RGB: Farbe A: Env. Mask	RGB: Dummy (Cubic Env)	

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

12 Datenbeeinflussung beim Exportieren

12.1 Grundsätzliche Funktionsweise

Bigex hat eine sehr mächtige Funktion integriert, die es erlaubt beim Exportieren in das IA/IGS Format **beliebige** Daten gezielt zu verändern. Diese **Export Daten Modifikationsfunktion**, im folgenden **EDM** genannt, arbeitet auf der Basis von Such- und Ersetzungsanweisungen. Diese Anweisungen werden aus mehreren benutzerdefinierten Textdateien eingelesen.

Grundsätzlich wird bei EDM unterschieden in Anweisungen zum Modifizieren von Objekteigenschaften und Anweisungen mit Aktionen zum verändern von 3D Modellen. Details zu beiden werden in den folgenden Kapiteln 12.3 Modifizieren von Objekteigenschaften und 12.4 Aktionsanweisungen erläutert.

Die *Bigex* EDM Funktion unterstützt reguläre Ausdrücke, was das uneingeschränkte Verändern von Werten in der IA/IGS Datei ermöglicht. Damit hat man die absolute Kontrolle über alle Werte in diesen Dateien und damit auch über das Aussehen und Verhalten eines Modells im Simulator selbst. Weiterhin ist damit der Weg frei für ein uneingeschränktes Experimentieren mit allen Shader Parametern.

Eingesetzt werden kann EDM um z.B. die Kurzform der Shadernamen durch die voll ausgeschriebene Form zu ersetzen oder um Pfade und Dateinamen für Texturbilder für z.B. dynamische Nummerntafeln zu verändern. Die Aktionsanweisungen können eingesetzt werden um z.B. mehrere Meshobjekte zu einem zu vereinigen. Dies wird z.B. benötigt wenn man mehr als 16 Materialien für ein Meshobjekt in Blender benötigt oder mehr als 256 Meshobjekte für ein Modell exportieren möchte.

Die Blender internen Datenstrukturen bleiben dabei in allen Fällen unverändert erhalten. Es werden ausschliesslich die exportierten Daten verändert.

Grundsätzlich können alle Daten beeinflusst werden, die in das IA/IGS Format exportiert werden. Das sind nicht nur Zeichenketten wie Shader- oder Objektnamen, sondern auch alle Zahlenwerte. So können z. B. die Fließkommazahlen der Umgebungslichtfarbe verändert werden oder die Positionen von Objekten.

Das Modifizieren der Daten erfolgt nach dem Suchen und Ersetzen Prinzip. Die *Bigex* EDM-Funktion unterstützt reguläre Ausdrücke nach der **Python Syntax** (Siehe [11]). Das heisst, der der Suchtext kann reguläre Ausdrücke enthalten und der Ersetzungstext kann entsprechende Referenzmarken auf passende Teile des Suchtextes enthalten.

Die Ausdrücke für das Suchen und Ersetzen liest *Bigex* aus je zwei benutzerdefinierten Textdateien für das IA und IGS-Format. Diese Textdateien werden im gewählten **Zielverzeichnis** für den IA/IGS Export erwartet. Sind diese Dateien dort vorhanden, werden sie von *Bigex* eingelesen und alle gefundenen und gültigen Such- und Ersetzungsausdrücke anschliessend auf die Exportdaten angewendet.

Gleich zu Beginn des Exports wird im Konsolenfenster, und falls eingeschaltet in der Log Datei, ausgegeben, welche Textdateien eingelesen und wie viele gültige Ausdrücke gefunden wurden. Im weiteren Verlauf des Exports wird dann auch detailliert ausgegeben welche Ersetzungen genau vorgenommen wurden.

Die beiden Textdateien mit den Ersetzungsanweisungen unterteilen sich in eine Datei mit festgelegtem Namen und eine mit einer festgelegten Erweiterung, die an den IA/IGS Dateinamen angehängt wird. Die Basisdatei mit dem festen Namen **IGS_ExpModFile.txt** bzw. **IA_ExpModFile.txt** kann benutzt werden um Ersetzungsausdrücke anzugeben, die für alle 3D-Modelle gleich sind. Das sind z.B. die Shadernamen.

Der Dateiname für die modellspezifische Textdatei setzt sich aus dem gewählten IA bzw. IGS Dateinamen und der festen Erweiterung **_IGSExpModFile.txt** bzw. **_IAExpModFile.txt** zusammen.

Beispiel: Wenn die gewählte IGS bzw. IA Exportdatei z.B. *MeineLok.igs* ist, dann wäre der Dateiname für diese Textdatei *MeineLok_IGSExpModFile.txt* bzw. *MeineLok_IAExpModFile.txt*

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

Diese EDM Dateien können verwendet werden um modellspezifische Änderungen durchzuführen. Das sind z.B. Zusammenfassen von Meshobjekten oder Setzen von speziellen Materialeigenschaften.

Vier Beispieldateien dazu sind im Unterverzeichnis **Bigex_ExampleObjects** des Installationspaketes enthalten.

12.2 Inhalt der Bigex Export Daten Modifikation (EDM) Dateien

Die Syntax der basis- und der modellspezifischen Textdatei sind gleich. Sie können Leerzeilen, Kommentarzeilen, Zeilen mit einer Suchbereichsangabe oder mit Such- und Ersetzungsausdrücken enthalten.

Hinweis: Kommentarzeilen haben als erstes Zeichen in der Zeile ein # und werden genau wie Leerzeilen von Bigex ignoriert.

12.3 Modifizieren von Objekteigenschaften

12.3.1 Suchbereich

Vor einem oder einer Gruppe von Such- und Ersetzungsausdrücken zum Ändern von Objekteigenschaften muss ein Suchbereich angegeben werden. Der Suchbereich setzt sich zusammen aus der Objektliste und der zu verändernden Eigenschaft der Listeneinträge.

Das folgende Kapitel 12.3.3 IGS- und IA-Objektlisten enthält eine vollständige Liste mit den Namen aller vorhandenen Objektlisten.

Hinweis: Zeilen mit Angaben zum Suchbereich haben als erstes Zeichen in der Zeile ein [, den Listennamen des Bereichs, einen Doppelpunkt :, den Namen der Eigenschaft eines Listeneintrags und als letztes Zeichen in der Zeile ein].

Alle folgenden Ersetzungsanweisungen bis zur nächsten Zeile die einen Suchbereich angibt, bzw. bis zum Dateiende, werden nur auf den Suchbereich angewandt, der vor diesen Anweisungen steht.

Beispiel:

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

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

Die ersten beiden Ersetzungsanweisungen in diesem Beispiel werden auf die *ShaderName* Eigenschaft aller Objekte in der Materials Objektliste angewendet.

Die dritte Ersetzungsanweisung in diesem Beispiel wird auf die Eigenschaft *Name* aller Objekte in der Objektliste *Objects* angewendet.

Möchte man tiefer geschachtelte Eigenschaften wie z.B. MipLODBias vom RenderStage1 eines Materials modifizieren, dann kann man diese durch anhängen einen Unterstrichs (_) gefolgt von dem Namen der Eigenschaft erreichen.

Beispiel1:

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

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

Beispiel 2:

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

Im ersten Beispiel wird der Wert der Eigenschaft *MipLODBias* vom *Render Stage 1* aller Materialien auf -1,0 gesetzt. Im zweiten Beispiel wird der Wert der Eigenschaft *PulseGlowConfig TargColor Green* nur für das Material mit Namen 'Mat1' auf 0,66 gesetzt.

Mehr dazu im folgenden Kapitel.

12.3.2 Such- und Ersetzungsanweisungen

Such- und Ersetzungsausdrücke bestehen aus zwei Teilen. Einem Suchtext, der am Anfang einer Zeile beginnt und einer Ersetzungsanweisung am Ende der selben Zeile. Beide sind getrennt durch das Zuweisungszeichen ('=').

Die Such- und Ersetzungsausdrücke unterstützen reguläre Ausdrücke nach der **Python Syntax** (Siehe [11]).

Eine Suchanweisung belegt eine Zeile in der Such- und Ersetzungsdatei.

Hinweis: Zeilen mit Such- und Ersetzungsausdrücken bestehen aus einer Such-Zeichenkette, die vom Beginn der Zeile bis zum = geht, dem Zuweisungszeichen = und einer Ersetzen-Zeichenkette, die vom = bis zum Zeilenende geht.

Hinweis: Alle Zeichen im Such- und Ersetzungstext, wie z.B.: Leerzeichen oder Punkt, gehören mit zum Such- bzw. Ersetzungstext. Es werden also alle Zeichen mit gesucht bzw. ersetzt. Ausgenommen hiervon sind die Steuerzeichen für reguläre Ausdrücke: '(', '[', '\ ' und das Zuweisungszeichen '='.

Für die Suchzeichenkette werden vier verschiedene Fälle unterschieden.

1. **Unbedingte Ersetzung:**
Wenn die Suchzeichenkette leer ist, also das Zuweisungszeichen = als erstes Zeichen in der Zeile steht, dann wird die im Suchbereich angegebene Eigenschaft **aller** Objekte der Objektliste durch die Ersetzungszeichenkette ersetzt, unabhängig davon welchen Wert die Eigenschaft vorher hatte.
2. **Eingeschränkt auf einen bestimmten Indexbereich von Objekten in einer Liste:**
Wenn die ersten Zeichen einer Suchzeichenkette **[n]** oder **[n-m]** sind, dann werden nur die **n-ten** bzw. die **n-ten bis m-ten** Objekte einer Objektliste durchsucht und nur der Wert deren Eigenschaft ersetzt.
3. **Eingeschränkt auf bestimmte Objekte einer Liste:**
Wenn die ersten Zeichen einer Suchzeichenkette **[Text]** sind, dann wird nur das Objekt mit dem Namen **Text** verändert. Alle anderen Objekte bleiben unverändert. Falls das Objekt keine Namen-Eigenschaft hat, wird dieser Such- und Ersetzungsausdruck ignoriert.
4. **Eingeschränkt auf bestimmte Inhalte von Eigenschaften:**
Wenn die Suchzeichenkette nicht mit = oder [beginnt, dann wird nach Werten von Eigenschaften gesucht, die **genau** der Suchzeichenkette entsprechen.

Alle diese Fälle können auch kombiniert werden.

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

Beispiele:

```

...
# Suchbereich: Objektliste Materials, Eigenschaft ShaderName
[Materials:ShaderName]
# 1. Ersetzt alle Shadernamen von allen Materialien durch SName2 unabhängig davon wie sie
vorher waren
=SName2

# 2. Ersetzt den Shadernamen von Material 99 in der Liste durch SName2 unabhängig davon wie er
vorher war
[99]=SName2

# 3. Ersetzt nur Shadernamen von Materialien durch SName2 wenn der Shadername SName1 ist
SName1=SName2

# 4. Ersetzt den Shadernamen von Material 99 in der Liste durch SName2, aber nur wenn dieser
SName1 ist
[99]SName1=SName2

# 5. Ersetzt den Shadernamen von Material 88 bis 99 in der Liste durch SName2, aber nur wenn
dieser SName1 ist
[88-99]SName1=SName2

# 6. Ersetzt den Shadernamen vom Material mit dem Namen MatName durch SName2, aber nur wenn
dieser SName1 ist
[MatName]SName1=SName2
...
[IGfMaterials:PulseGlowConfig_TargColor_Green]
# 7. Setzt bei dem Material mit dem Namen 'Mat88' den Wert der Eigenschaft
# PulseGlowConfig TargColor Green auf 0,75
[Mat88]=0.75
...

```

Die vier EDM Musterdateien **IA** bzw. **IGS_ExpModFile.txt**, **IA** bzw. **IGS_specific_IGSExpModFile.txt** im Unterverzeichnis **ExampleObjects** des Installationspaketes enthalten weitere Beispiele für Such- und Ersetzungsausdrücke.

12.3.3 IGS- und IA-Objektlisten

Objekte in den folgenden Objektlisten können beim Export in das IGS Format (IGf) beeinflusst werden:

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

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

Die Objekte in den folgenden Objektlisten können beim Export in das IA Format (IAf) beeinflusst werden:

- IAfHeaders
- IAfTriggers
- IAfAnimationNodes
- IAfMotionControllers
- IAfMotionSets

Wenn diese Objektlistenamen in Suchbereichen der EDM Funktion verwendet werden, dann kann die führende Kennzeichnung **IAf** bzw. **IGf** auch weggelassen werden.

Beispiel:

```
# Suchbereich: Materials Objektlist, Eigenschaft ShaderName
[IGfMaterials:ShaderName]
...
# ist identisch zu:
[Materials:ShaderName]
...
```

12.3.4 Eigenschaften von Objekten

Die einzelnen Objekte in diesen Objektlisten haben eine Vielzahl von unterschiedlichen Eigenschaften. Insgesamt gibt es mehrere Hundert Eigenschaften. Um die genaue Bezeichnung einer Eigenschaft herauszufinden, die man ersetzen möchte, empfiehlt es sich, einen Exportdurchlauf **ohne** Ersetzen zu machen. Anschließend kann man dann den exakten Namen der Eigenschaft die man verändern möchte aus der Log Datei herauskopieren.

Beispiel: Ersetzen von Objektnamen

Auszug aus einer Log Datei von einem Export **ohne** Ersetzung:

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

Bei einem Export mit der folgenden Ersetzungsanweisung

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

wird dann anstelle des Objektnamens **AB_FSSeitFenst_Rg** in Blender der Name aus der Ersetzungsanweisung **1_0500_Seitenfenster_Rechts** in die IGS Datei geschrieben.

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

Auszug aus der in diesem Fall erzeugten Log Datei **mit** Ersetzung:

```

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

```

Weitere Beispiele für Ersetzungsanweisungen sind in den zwei Beispieldateien **IGS_ExpModFile.txt** und **Bigex_ExampleObject_1_IGSExpModFile.txt** im Unterverzeichnis **Bigex_ShaderTest** des Installationspaketes enthalten.

12.4 Aktionsanweisungen

Neben den Such- und Ersetzungsanweisungen gibt es eine weitere Gruppe von Anweisungen, die Aktionsanweisungen. Mit diesen lassen sich automatisiert Veränderungen am 3D Modell selbst durchführen.

Aktionsanweisungen beginnen auch mit einer Bereichsangabe, gefolgt von einer Aktion.

Hinweis: Zeilen mit Bereichs- und Aktionsangaben haben als **erstes Zeichen in der Zeile** ein [, den Listennamen des Bereichs, zwei Doppelpunkte ::, den Namen der Aktion und als **letztes Zeichen in der Zeile** ein].

Beispiel:

```

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

```

Derzeit gibt es nur eine Aktionsanweisung.

12.4.1 Zusammenfassen von Meshobjekten

Mit der Aktion Meshobjekte zusammenfassen (*IGfObjects::Merge*) können ein oder mehrere Objekte beim Exportieren zu einem einzigen Objekt vereinigt werden. Dabei werden alle Vertices, Flächen, Materialien und UV-Koordinaten mit zum neuen, vereinigten Meshobjekt übernommen. Objekthierarchien werden soweit möglich erhalten, bzw. für die gelöschten Objekte sinnvoll zusammengeführt.

Die Aktionsanweisungen für das Zusammenführen von Meshobjekten beginnt mit der Bereichsangabe und der Aktion: *Merge*. In den darauffolgenden Zeilen bis zur nächsten Bereichsangabe, bzw. dem Dateiende, folgen dann ein oder mehrere Anweisungen für das Zusammenführen. Sie beginnen mit dem Namen des neuen Objekts, das später alle Daten der zusammenzuführenden Objekte enthält, gefolgt von einem Gleichheitszeichen =. Es folgen ein Objektname oder eine durch Kommas getrennte Liste mit den Namen aller zusammenzuführenden Meshobjekte.

Wenn das Objekt in dem die anderen Objekte zusammengeführt werden sollen nicht existiert und mehrere zusammenzuführende Objekt gelistet wurden, dann nimmt das erste Objekt dieser Liste eine Sonderstellung ein. Es wird das Ursprungsobjekt an das die anderen Objekte angehängt werden.

Wenn die Liste mit zusammenzuführenden Objekten Namen von nicht existierenden Objekten enthält, dann werden diese ignoriert und nur alle existierenden Objekte zusammengeführt.

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

Beispiele:

```
...
# Merge Aktion Bereichs- und Aktionsangabe
[IGfObjects::Merge]

# Erstelle das neue Objekt MergedObjs aus Obj1 und Obj2
MergedObjs=Obj1,Obj2

# Hänge Obj2 an Obj1 an
Obj1=Obj1,Obj2
# genau das selbe
Obj1=Obj2

# Erstelle das neue Objekt MergedObjs aus Obj1, Obj2, Obj3 und Obj4
MergedObjs=Obj1,Obj2,Obj3,Obj4
# genau das selbe mit regulärem Ausdruck
MergedObjs=Obj[1-4]
# genau das selbe mit anderem regulärem Ausdruck
MergedObjs=Obj[4321]
...
```

Die objektspezifische Beispieldatei ***IGS_specific_IGSExpModFile.txt*** im Unterverzeichnis ***Bigex_ExampleObjects*** des Installationspaketes enthält weitere Beispiele für die Aktion *Merge*.

Hinweis: Es ist sinnvoll Mesh Objekte zusammenzufassen, die sich auf der selben *Level of Distanz* (LOD) Ebene befinden. Diese Objekte werden später bei der Umwandlung vom *IGS* in das *.GeoPcDx* Format, die der Rail Works Asset Editor durchführt, sowieso zu einer Gruppe zusammengefasst.

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

13 Bigex im Kommandozeilen Modus laufen lassen

Neben der Nutzung als Blender IA/IGS Exporter kann man das *Bigex* Python Skript auch zur Analyse von vorhandenen IA/IGS verwenden. Dieser Betriebsmodus kann sehr einfach von einer Kommandozeile in einem Konsolenfenster aus gestartet werden.

Der Analysator Modus erlaubt es den Inhalt von vorhandenen IA/IGS Dateien im Konsolenfenster oder optional in eine Log Datei auszugeben. Weiterhin ist es möglich sich eine Datei mit Komma separierten Werten (csv) mit den IA/IGS Daten erzeugen zu lassen. Dies kann dann sehr einfach in ein Tabellenkalkulationsprogramm wie Openoffice Calc oder Microsoft Excel geladen werden. Dort kann man dann alle Daten genauer analysieren und weiterverarbeiten.

Das Verhalten von *Bigex* im Analysator Modus wird über Optionen beim Aufruf auf der Kommandozeile gesteuert.

Bigex von einer Kommandozeile aus zu starten ist sehr einfach. Man braucht nur das Skript ***bigex25.pyc***, wenn man Python 2.5 benutzt, bzw. ***bigex26.pyc***, wenn man Python 2.6 benutzt, aufzurufen.

Unter Linux, braucht man nur über das Start Menü ein neues Kommandozeilenfenster zu öffnen und das folgende Kommando einzugeben:

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

Unter Windows, öffnet man über das Start Menü ein neues Kommandozeilenfenster (Start => Run: "cmd" => OK) und gibt dann das folgende Kommando ein:

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

Möglicherweise muss der Pfad zur ausführbaren Python Interpreter Datei noch angepasst werden. Die Blender Skripte liegen üblicherweise im Verzeichnis:

```
<Programme Laufwerk>:\<Programme Verzeichnis>\<Blender Installationsverzeichnis>\.blender\scripts
also z.B.: C:\Programme\Blender\.blender\scripts
```

Nach erfolgreichem Start mit der Option '`--help`' wird der folgende Hilfetext ausgegeben:

HenningBR218	Blender IA/IGS Exporter (Bigex) Handbuch	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
```

Wenn eine Option angegeben wird, die eine Ausgabedatei anlegen soll, dann wird diese im selben Ordner erstellt in dem auch die IA/IGS Quelldatei liegt.

Einige Beispiele für den Skriptaufruf:

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

gibt nur den Dateikopf und eine kurze Übersicht vom Inhalt von *MyIGSfile.igs* aus. Damit erhält man schnell einen Überblick über die IGS Datei.

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

gibt den kompletten Inhalt von *MyIGSfile.igs* aus, einschließlich aller Details aller Objekte. Zusätzlich wird noch eine Zusammenfassung und die benötigte Zeit für alle einzelnen Schritte ausgegeben.

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

gibt eine Liste aller Objektlisten von *MyIGSfile.igs* aus.

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

gibt den kompletten Inhalt von *MyIGSfile.igs* aus. Zusätzlich wird für jede Objektliste, die mindestens ein Objekt enthält, eine Datei mit Komma separierten Werten (csv) ausgegeben.

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

14 Problemlbericht senden

Keine Software ist ohne Fehler. Deswegen kann man, wenn das Export Skript z.B. mit einer Fehlermeldung abbricht, einen Fehlerbericht an die in Abschnitt 5 *Copyright, Lizenz, Haftung, Kontakt, Dank* angegebene Mailadresse senden. Dies hilft bei der Verbesserung von *Bigex*.

Folgende Punkte müssen erfüllt sein bevor man eine Fehlermeldung abschickt:

- Entsprechen die benutzten Programme der Installationsanleitung in diesem Handbuch?
- Lässt sich der Fehler mindestens dreimal reproduzieren?
- Sind aus dem Blender Modell alle Objekte entfernt worden, die das Problem **nicht** verursachen?
- Sind die kritischen Objekte noch im Blender Modell?
Dies sind entweder Objekte bei denen der Export mit einer Fehlermeldung im Konsolenfenster abbricht oder Objekte, die unvollständig oder garnicht exportiert werden.

Um eine Ursachenanalyse durchführen zu können, müssen die folgenden Daten in einer ZIP Datei vorliegen:

- Ausgabe des Konsolenfensters vom kompletten Exportvorgang der das Problem verursacht.
- Log Datei, die mit in der Benutzeroberfläche eingeschalteter Option *Verbose* erzeugt wurde.
- Blender .blend Datei, die nur die Objekte enthält, die den Fehler auslösen.

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

15 Anhang

15.1 Abbildung der Blender Daten auf das IA/IGS Format

Beim Exportieren in das IA/IGS-Format müssen die 3D-Daten von Blender in die Datenfelder der einzelnen IA/IGS-Objekte übertragen werden. Da Blender aber sehr viele Regler, Eingabefelder und Knöpfe besitzt, ist es nicht immer offensichtlich woher die Werte kommen, die in die einzelnen Datenfelder in der IA/IGS-Datei eingetragen werden.

Wenn man in Blender in einem bestimmten Eingabefeld z.B. den Namen einer Texturdatei setzt, dann kann man diesen Dateinamen nach dem Export in die IGS-Datei auch an der entsprechenden Stelle in der Log-Datei wiederfinden.

Die folgenden Tabellen zeigen welche Eingabefelder von Blender beim Exportieren in das IA/IGS-Format auf welche Datenfelder in der IA/IGS-Datei abgebildet werden. Datenfelder, die nicht in der Tabelle stehen können mit der *Bigex* EDM Funktion beim Export gesetzt werden.

Die linke Tabellenseite mit der Überschrift **Blender** zeigt den kompletten Pfad zum Eingabefeld in Blender. Ausgehend vom Unterfenster in Blender über das Panel und den Reiter bis zum Namen des Feldes/Schiebereglers auf dem Reiter.

Im mittleren Tabellenteil **IGS-Datei** bzw. **IA-Datei** wird der Pfad zum Dateneintrag in der IA/IGS Datei aufgelistet. Ausgehend von der Objektgruppe bis zum Namen der Eigenschaft, für die der Wert gesetzt werden soll.

Auf der rechten Tabellenseite stehen noch einige Kommentare sowie der Aktivierungsstatus des Eintrages.

Hinweis: Alle Datenfelder der IA/IGS Datei, die nicht in den Tabellen aufgeführt sind, können nicht über Blender gesetzt werden. Sie werden mit sinnvollen festen Werten belegt. Aber wie alle anderen IA/IGS Datenfelder so können auch diese natürlich mit der *Bigex* EDM Funktion beim Export gesetzt werden.

15.1.1 Blender Eingabefelder für das IGS Format

Übersicht der Blender Eingabefelder für das IGS Dateiformat:

Blender				IGS Datei		Kommentar	Status
Fenster	Panel	Tab	Feld	IGS Objekt-liste	Eigenschaft		Aktiv
Button Window	Shading Material Buttons (F5)	Links and Pipeline	MA:	Materials	Name		ja
Button Window	Shading Material Buttons (F5)	Texture	Erster aktiver Textur Channel Name	Materials	ShaderName		ja
---	---	---	---	Materials	RenderStageCount	Automatisch berechnet: Anzahl der aktiven Texture Layer - 1.	ja

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

Blender				IGS Datei		Kommentar	Status
Fenster	Panel	Tab	Feld	IGS Objekt-liste	Eigenschaft		Aktiv
Button Window	Shading(F5)	Shaders	LBias	Materials	RSx.MipLODBias	Der Shading Wert wird für alle aktiven Render Stages gesetzt.	nein (Nicht via Python auslesbar)
Button Window	Scene(F10)	Format	FPS	Materials	RSx.FPS	Der Scene Wert wird für alle aktiven Render Stages gesetzt.	ja
Scripts Window	<i>Bigex</i> GUI	Material	RSxArgy	Materials	RSx.UVArguments	Das IGS Format nimmt nur die ersten 6 der 8 Argumente der <i>Bigex</i> GUI auf.	ja
Button Window	Shading(F5)	Links and Pipeline	Zoffs	Materials	ZBias	Wert wird auf Ganzzahl gerundet.	ja
Scripts Window	<i>Bigex</i> GUI	Material	Viewer Facing	Materials	ViewFacing		ja
---	---	---	---	Materials	AmbientColor_Red AmbientColor_Green AmbientColor_Blue AmbientColor_Alpha	Alle mit 1,0 vorbelegt. Kann mit EDM überschrieben werden.	ja
Button Window	Material Buttons	Material	Col RGB	Materials	DiffuseColor_Red DiffuseColor_Green DiffuseColor_Blue		ja
Button Window	Material Buttons	Material	Mir RGB	Materials	EmissiveColor_Red EmissiveColor_Green EmissiveColor_Blue		ja
Button Window	Material Buttons	Shaders	Emit	Materials	EmissiveStrength		ja
Button Window	Material Buttons	Material	Spe RGB	Materials	SpecularColor_Red SpecularColor_Green SpecularColor_Blue		ja
Button Window	Material Buttons	Material	A	Materials	DiffuseColor_Alpha SpecularColor_Alpha EmissiveColor_Alpha		ja
Button Window	Material Buttons	Shaders	Spec	Materials	SpecularPower	Um einen größeren Wertebereich für SpecularPower zu erhalten, werden die beiden Blender Felder multipliziert.	ja
Button Window	Material Buttons	Shaders	Hard				
---	---	---	---	Headers	ObjectCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfObjects'	ja
---	---	---	---	Headers	MeshCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfMeshes'	ja

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

Blender				IGS Datei		Kommentar	Status
Fenster	Panel	Tab	Feld	IGS Objekt-liste	Eigenschaft		Aktiv
---	---	---	---	Headers	BoneCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfBones'	ja
---	---	---	---	Headers	MaterialCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfMaterials'	ja
---	---	---	---	Headers	LightCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfLights'	ja
Button Window	World Buttons	World	AmbR AmbG AmbB	Headers	AmbientColor_Red AmbientColor_Green AmbientColor_Blue		ja
---	---	---	---	Headers	AmbientColor_Alpha	Fest auf 0,0 gesetzt.	ja
---	---	---	---	Headers	SplineCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfSplines'	ja
---	---	---	---	Headers	GenericItemCount	Automatischberechnet: Anzahl der Objekte in Objektliste 'IGfGenericItems'	ja
3D View	---	Transform Properties	Vertex X Vertex Z Vertex Y	Verts	Point_X Point_Y Point_Z		ja
---	---	---	---	Verts	Point_Weight	Mit 1,0 vorbelegt. Kann mit EDM überschrieben werden.	ja
3D View	---	Weight Paint Properties	Weight	Verts	Normal_X Normal_Y Normal_Z	Die Zahlenwerte sind aus den Blender internen Datenstrukturen: Normale für diesen Vertice	ja
---	---	---	---	Verts	Normal_Weight	Mit 1,0 vorbelegt. Kann mit EDM überschrieben werden.	ja
---	---	---	---	Verts	ValidUVsCount	Automatisch berechnet: Anzahl der UV Layer	ja
U/V Image Editor	---	---	---	Verts	UV00, ..., UV07	Die Zahlenwerte sind aus den Blender internen Datenstrukturen: UV-Koordinaten für diesen Vertice	ja
3D View	Editing (F9)	Mesh Tools More	Draw Normals	Triangles	Normal_X Normal_Y Normal_Z	Die Zahlenwerte sind aus den Blender internen Datenstrukturen: Flächennormale der Fläche	ja

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

Blender				IGS Datei		Kommentar	Status
Fenster	Panel	Tab	Feld	IGS Objekt-liste	Eigenschaft		Aktiv
---	---	---	---	Triangles	Normal_Weight	Mit 1,0 vorgelegt. Kann mit EDM überschrieben werden.	ja
---	---	---	---	Meshes	VerticeCount	Automatisch berechnet: Anzahl der Vertices von Mesh	ja
---	---	---	---	Meshes	TriangleCount	Automatisch berechnet: Anzahl der Triangles von Mesh	ja
3D View	Viewport Shading (Z, Shift-Z, Alt-Z)	---	Bounding Box	Meshes	BoundingBox_ LowerLeftPointX LowerLeftPointY LowerLeftPointZ LowerLeftPointScale UpperRightPointX UpperRightPointY UpperRightPointZ UpperRightPointScale	Automatisch berechnet: Maximale Ausmaße des Mesh.	ja
Button Window	Object (F7)	Object and Links	OB:	Objects	Name		ja
---	---	---	---	Objects	NameUniqueID	Automatisch berechnet: Index des Objekts in der Objektlste 'IGfObjects'	ja
---	---	---	---	Objects	MeshLODCount	Automatisch berechnet: Anzahl der Mesh LODs vom Objekt	ja
---	---	---	---	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	Die 16 Zahlenwerte sind aus den Blender internen Datenstrukturen: 4x4 Transformations Matrix (TM) vom Objekt	ja
Outliner	---	---	Object Tree	Objects	ChildrenCount	Automatisch berechnet: Anzahl der Untergeordneten Objekte in der Objekthierarchie	ja

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

Blender				IGS Datei		Kommentar	Status
Fenster	Panel	Tab	Feld	IGS Objekt-liste	Eigenschaft		Aktiv
Button Window	Texture Buttons (F6)	Image	Image / Move File name	TextureNames	[0], ..., [N]	Automatisch berechnet: Material1, Textur Layer1 Material1, Textur Layer2 ... Material1, Textur LayerN Material2, Textur Layer1 ... Material2, Textur LayerN ...	ja

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

15.1.2 Blender Eingabefelder für das IA Format

Übersicht der Blender Eingabefelder für das IA Dateiformat:

Blender				IA Datei		Kommentar	Status
Fenster	Panel	Tab	Feld	IA Objekt Liste	Eigenschaft		Aktiv
Button Window	Scene(F10)	Anim	Sta: End:	Headers	TotalFrameCount	Automatisch berechnet: Endframe - Startframe +1	ja
Button Window	Scene(F10)	Format	FPS:	Headers	SampleRate	Automatisch berechnet: Um den Wertebereich zu erhöhen werden die Eingabefelder FPS: und /: (Frames per second base) multipliziert.	ja
Button Window	Scene(F10)	Format	/: (Frames per second base)				
---	---	---	---	Headers	AnimationNodeCount	Automatisch berechnet: Anzahl der Objekte in der Objektliste 'AnimationNodes' (Ist gleich der Anzahl der animierten Meshobjekte)	ja
Button Window	Objects (F7)	Object and Links	OB:	Animation Nodes	Name	Name des zugehörigen Meshobjekts	ja
---	---	---	---	MotionSets	MotionVector4FrameNN_X MotionVector4FrameNN_Y MotionVector4FrameNN_Z	Automatisch berechnet: Position des animierten Meshobjekts in FrameNNN	ja
---	---	---	---	MotionSets	MotionQuaternionFrameNNN_a MotionQuaternionFrameNNN_b MotionQuaternionFrameNNN_c MotionQuaternionFrameNNN_d	Automatisch berechnet: Rotation des animierten Meshobjekts in FrameNNN	ja

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

15.2 Verweise

- [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> oder <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