SEGA™

Katana
Ninja Utilities

Katana Ninja Utilities
Table of Contents

01. PowerVR Texture Converter

How to Use PVRConv

PVRConv is a command line tool which converts various bitmap graphic formats into PowerVR textures.

The following input formats are supported by PVRConv:

- .bmp: PC bitmap file (24-bit)
- .pic: Softimage texture
- .tga: Targa file, used by Light Wave 3D, 3D Studio MAX
- .pix,
- .mask: Alias texture with optional alpha mask..

⚠ **NOTE:** The conversion core of PVRConv is called internally by each of the Ninja 3D model export tools/plug-ins.

Command Line Usage

```
PVRConv [Options] <InputFile> [<AlphaFile>]
Where: InputFile = .bmp .pic .tga .pix
AlphaFile = .bmp .mask
Output: InputFile.pvr
```

To receive usage information from the command line, enter PVRConv with no options or parameters.

⚠ **NOTE:** The optional `outpath` parameter listed in the command line help text currently does not work. Use the -p option instead.

Alpha Channel

Two of the supported bitmap formats (.bmp, .pix) do not store alpha channel information internally. To create a PowerVR texture with alpha values for these formats, PVRConv allows a second bitmap to be supplied on the command line.

.bmp:

Start with a source bitmap with an alpha channel. Save the RGB image data as a 24-bit .bmp file. Save the alpha channel as a separate 24-bit bitmap. The red components of the RGB data in this second file will be used as the alpha input for PVRConv. For example: PVRConv Texture.bmp Alpha.bmp -> Texture.pvr

.pix:

When exporting textures from Alias, the RGB texture data is stored in a .pix file and the alpha channel is saved in a .mask file. Use both of these files as input for PVRConv. PVRConv Texture.pix Alpha.mask -> Texture.pvr PVRConv requires an alpha channel for .bmp and .pix files only when generating ARGB_4444 and ARGB_1555 PowerVR textures.

PVRConv Defaults

When no options are specified, PVRConv checks the input file's dimensions and any alpha information and generates an output texture based on the following defaults:

Square texture:	TWIDDLED_MM (Twiddled with auto mip-map generation)
Rectangular texture:	RECTANGLE (No mip-maps allowed)
No alpha channel:	RGB_565
Alpha channel (0-255):	ARGB_4444
Alpha of only 0 & 255:	ARGB_1555
Stride texture:	OFF
Mip-map bank:	BANK_0
Dither:	ON
Dither alpha channel:	ON
Output path:	Current directory
Check file suffix:	ON
Global index:	OFF

PVRConv Options

The following options may be supplied on the command line to force output of a specific texture type or override the default operation of PVRConv.

-p <dir> | -path <dir> :

Specify an output directory. Path names are relative to the current folder unless a full 'drive:\directory' path is provided.

```
PVRConv -p my textures Texture1.tga
PVRConv Texture2.bmp -p c:\converted\batch2
```

-ns | -nosuffix :

Ignore input file suffix. This option is only necessary for files with endings other than .bmp, .pic, .tga, .pix, and .mask.

⚠ **NOTE:** Supported graphics formats are auto-detected and converted even if a suffix is not present.

```
PVRConv -ns Texture
```

-gi <num> | -globalindex <num> :

Specify a Global Index. A GBIX chunk containing the provided 32-bit <num> is written to the head of the output file. See 'PowerVR File Format' for more info. This index is required by Ninja applications.

```
PVRConv -gi 99 Texture.bmp
```

-t | -twiddled :

Force twiddled output. PowerVR twiddled format rearranges the order of pixels in a texture for optimal rendering performance. Twiddled textures are displayed much faster than rectangle or stride textures.

Square textures are automatically converted to twiddled. When this option is specified, rectangular textures are expanded to the nearest square size and padded with blank space. For example:

```
128x32 -> 128x128 -> Twiddled Texture
```

-r | -rectangle :

Force rectangle texture output. Rectangle textures are stored in standard linear order but yield poorer performance than twiddled textures. Rectangle dimensions must be powers of two.

When this option is specified square textures are written in rectangle format.

⚠ **NOTE:** Rectangle textures can not have mip-maps.

-s | -stride :

Force stride texture output. Stride textures have the same linear arrangement as rectangle textures but allow the width to be varied by multiples of 32 pixels instead of powers of two.

Stride textures are most commonly used to take a portion of the frame buffer and re-render it as a texture

-nm | -nomipmap :

Turn off mip-map auto-generation for twiddled textures. Mip-maps are not created for rectangle or stride textures.

-nd | -nodither :

Specify no dithering when performing color reduction of RGB data.

-na | -noadither :

Specify no dithering of alpha values when performing color reduction.

-5 | -565 :

Force RBG_565 texture format. Alpha channels are ignored if present.

The remaining texture formatting options require alpha channel data to be present in the source bitmap or provided as a second input for .bmp and .pix files:

-4 | -4444 :

Force ARGB_4444 texture format. Alpha values in the source texture range from 0 to 255 and are converted directly to PowerVR format.

-1 | -1555 :

Force ARGB_1555 texture format. Alpha values in the source texture are either 0 or 255 and are written to the destination texture as either 0 or 1 respectively.

-t1 | -t1555 :

Force ARGB_1555 texture format from translucent source data. Alpha values in the source texture range from 0 to 255 and are truncated to either 0 or 1 based on <= 127 or > 128.

-b4 | -b4444 :

Force ARGB_4444 texture format from bi-level alpha data. Alpha values in the source texture are either 0 or 255 and are converted directly to PowerVR format.

-ra | -ralpha :

Reverse alpha values. Reverse Alpha= 255 - Alpha.

PowerVR Texture Format

PowerVR texture files (.pvr) are based on the standard IFF chunk file format. Textures contain a PVRT data chunk and an optional GBIX (Global Index) chunk for Ninja applications.

The PVRT chunk contains information about the texture type (i.e. mip-mapped, twiddled, VQ compressed, etc.), width, and height.

The GBIX chunk contains a global index number used by Ninja to reference the texture. This number can be supplied by PVRConv and/or modified using the GIGen tool.

Table 1: PVR Texture File Map

Size	Name	Sample Data
DWORD	Chunk	`GBIX`
DWORD	NextPtr	0x4 (CurrentPos + 4 bytes -> PVRT chunk)
DWORD	GlobalIndex	0x63
DWORD	Chunk	`PVRT`
DWORD	NextPtr	0x2AAB4 (EOF - CurrentPos)
DWORD	TextureType	0x102 (TWIDDLED | ARGB_4444)
WORD	Width	256
WORD	Height	256
BYTE	Data[]	0..EOF (DataLen = Width * Height * Depth)
	.	.
	.	.

⚠ **NOTE:** Width/Height values must be 8, 16, 32, 64, 128, 256, 512, 1024.

Table 2: Texture Types

Texture Type	Value		Color Type	Value
TWIDDLED	0x0100		ARGB_1555	0x00
TWIDDLED_MM	0x0200		RGB_56	0x01
VQ	0x0300		ARGB_4444	0x02
VQ_MM	0x0400		YUV_422	0x03
PALETTIZE4	0x0500		BUMP	0x04
PALETTIZE4_MM	0x0600		RGB_555	0x05
PALETTIZE8	0x0700			
PALETTIZE8_MM	0x0800			
RECTANGLE	0x0900			
RECTANGLE_MM	0x0A00			
STRIDE	0x0B00			
STRIDE_MM	0x0C00			
TWIDDLED_RECTANGLE	0x0D00			
ABGR (32-bit)	0x0E00			
ABGR_MM (32-bit)	0x0F00			

SEGA™

02. VQ Compression Tool

How to Use VQ Compress

VQ Compress is a command line tool which converts PC bitmap graphic files into PowerVR VQ compressed textures.

Input files must be:

.bmp: PC bitmap file (24-bit)

⚠ **Note:** VQ Compress is pre-alpha software and is provided for evaluation purposes. This tool will be replaced by VQ compression functionality in PVRConv.

VQ Compress has the following limitations:

- No Softimage, Alias, or Targa file support.
- No Rectangle or Stride texture support.
- Source alpha channels ignored.
- No mip-map generation.
- No user-specified image quality parameters (dithering, etc.)
- No “Sliding-window” VQ compression support.

Command Line Usage

VQ [Options] <InputFile> [<SampleOut>]

Where: InputFile = Input filename (.bmp)

SampleOut = Optional output file (.bmp) to sample results

Output: InputFile.pvr

[SampleOut.bmp]

To receive usage information from the command line, enter VQ with no options or parameters.

VQ Compress Defaults

When no options are specified, VQ Compress checks the input file's dimensions and generates an output texture with the following defaults:

Texture format: VQ (VQ compressed with no mip-map)

Pixel format: ARGB_1555

Dither: ON

Output path: Current directory

Check file suffix: ON

Global index: OFF

VQ Compress Options

The following options may be supplied on the command line to force output of a specific texture type or override the default operation of VQ Compress:

-g <number> :

Specify a Global Index. A GBIX chunk containing the provided 32-bit <number> is written to the head of the output file.

See the "*PowerVR Texture Format*" section of PVRConv for more info.

This index is required by Ninja applications.

Example:

```
VQ -g 99 Texture.bmp
```

-5 : Force RBG_565 texture format.

-4 : Force ARGB_4444 texture format. Alpha set to 0.

This option can be used to evaluate VQ compression quality of 4-bits-per-channel RGB data.

-1 : Force ARGB_1555 texture format (default). Alpha set to 0.

VQ Compression Ratios

VQ does not always generate PowerVR textures that are smaller than the original bitmap. This is due to the nature of VQ compression, which requires a 2k codebook regardless of texture size. A detailed description of the VQ compression algorithm is provided in the "*Texture*" section of the Kamui documentation.

The following tables illustrate the compression ratios of supported texture formats:

Texture Size	Original	Compress	Compress Ratio
8x8	128	2064	16.125
16x16	512	2112	4.125
32x32	2048	2304	1.125
64x64	8192	3072	0.375
128x128	32768	6144	0.1875
256x256	131072	18432	0.140625
512x512	524288	67584	0.128906
1024x1024	2097152	264192	0.125977 (8:1)

Table 1: VQ Compression Ratio of Textures

Texture Size	Original	Compress	Compress Ratio
8x8 MM	169	2069	12.242603
16x16 MM	681	2133	3.13259
32x32 MM	2729	2389	0.875412
64x64 MM	10921	3413	0.312517
128x128 MM	43689	7509	0.171874
256x256 MM	174761	23893	0.136718
512x512 MM	699049	89429	0.127930
1024x1024 MM	2796201	351573	0.125732 (8:1)

Table 2: VQ Compression Ratio of Textures w/ Mip-map

03. Using the LightWave Converter

Overview of the LightWave Converter

The LightWave converter is used to convert model data that was originally created by LightWave into model and motion files (.nja, .nam, pvr) for Ninja. These files are then used with the Ninja library for model display and motion operations.

Caution: See "*Restrictions*" section of this document for important information. LightWave Converter is pre-alpha software and is provided for evaluation purposes.

Output Files

LightWave outputs several different types of files. These are explained below.

1) MRS (motion resource) files

Model hierarchical structure data files. These ASCII files contain data concerning model hierarchical structures, movement, rotation, enlargement, etc.

2) PVR files

Texture files.

3) NJA files

Ninja model files. These ASCII files contain model structures.

4) NAM files

Ninja motion files. These ASCII files contain motion structures.

5) NAS files

Ninja shape files. These ASCII files contain shape structures.

6) NSC files

Ninja scene files. These ASCII files contain path information and NJA, NAM, and NAS file names. The user can easily change these files.

Path names can be specified either as absolute or relative; do not specify a path name if the path is for the current directory.

⚠ **Note:** When a user creates an NSC file, it is possible that the global indexes for textures could be duplicated. In this event, execute GIGEN.EXE to make the global indexes for textures sequential.

Execution

The LightWave Converter is executed from the MS-DOS command prompt. Type the following on the command line.

```
lwtonj  -i<input file> [-o<output file>]
lwtonj  <input file> [<output path>]

   Input file: .lws
```

Executing the above command creates .nja, .nam, .mrs, pvr, nas, nsc files.

The `lwtonj` command options are explained below.

-i or -in <infile> : Set In file

Sets input files.

-o or –out <outpath > : Set out path

Sets the output file path.

-tp or -texpath <texpath> : Set texture path

Sets the input texture path. This option disables the texture path which has been set in Lwo and enables the path set by this option.

-inm or -inmrs<Infile> : Set In mrs file

Specifies an MRS (motion resource) file to be input. If this option is specified, motion is output in terms of the difference from MRS.

-a or -trr : Top : Trans Rot, others : Rot Only (type A)

Outputs the motion file as Type A.

With this type, only the root node has `trans`; only `rot` is output for other nodes.

-b or –trt : All : Trans Rot (type B)

Outputs the motion file as Type B.

With this type, all the nodes have `trans` and `rot`.

-c or -trs : All : Trans Rot Scale (type C)

Outputs the motion file as Type C.

With this type, all the nodes have `trans`, `rot`, `scale`.

-ar or –arot : Rotation key all

If the amount of change in the rotation component of motion between frames is constant, the key frame for that portion is cut.

If this option is specified, all key frames are output.

-as or –ascale : Scale key all

If the amount of change in the scale component of motion between frames is constant, the key frame for that portion is cut.

If this option is specified, all key frames are output.

-at or –atrans : Translation key all

If the amount of change in the translation component of motion between frames is constant, the key frame for that portion is cut. If this option is specified, all key frames are output.

-ft or –forcetex : Force texture convert

If this is checked, if a texture with the same name exists in the output destination, that texture is overwritten.

Normally, if a file with the same name exists in the output destination, that file is not converted.

-g or –gibase : Global index base

The starting number of the texture globalIndex can be set by inputting a number value here.

The default value is `0`.

-km or -keymotion : Out keyframe motion

Outputs motion data in key frames.

-m or -mrs : Out mrs file

Outputs an MRS file.

Outputs an MRS file on the same path as NJA.

-na or -noauto : No auto search path

Disables the automatic path search function of the converter. In case that Lwo files and texture files are not found in the specified directory, those files are searched automatically in the same drive by this function. This function is ignored by this option.

-nmd or -nomodel : No out model file (.nja)

Disables output of NJA files.

-nmt or -nomotion : No out motion file (.nam)

Disables output of NAM files.

-nt or -notex : No texture

Disables textures. This option disables the texture data even if it is in the model.

-pn or –polynormal : Print polygon normal

Outputs the polygon normal line to an .NJA file.

-pr or –prot : Print rotation

If the rotation values and motion data included in each object in an OBJECT tree are the same and constant, the rotation motion for that node is NULL. If this option is specified, all data is output.

-ps or –pscale : Print scale

If the scale values and motion data included in each object in an OBJECT tree are the same and constant, the scale motion for that node is NULL. If this option is specified, all data is output.

-pt or –ptrans : Print translation

If the translation values and motion data included in each object in an OBJECT tree are the same and constant, the translation motion for that node is NULL. If this option is specified, all data is output.

-S or -Strip : Use strip convert

Converts a polygon with 3 vertices to a strip polygon. A strip polygon is a continuous polygon.

-sp or -shape : Out shape file (.nas)

Outputs shape files (.nas) when there is shape data. Details on shapes are provided later.

-St : 4 to 3 convert

Converts a polygon with 4 vertices to a polygon with 3 vertices and converts the polygon with 3 vertices, including the converted polygon, to a strip polygon.

-ST : N to 3 convert

Converts a polygon with N vertices to a polygon with 3 vertices and converts the polygon with 3 vertices, including the converted polygon, to a strip polygon.

-STt : 4 & N to 3 convert

Converts polygons of 4 and N vertices to a polygon with 3 vertices and converts the polygon with 3 vertices, including the converted polygon, to a strip polygon.

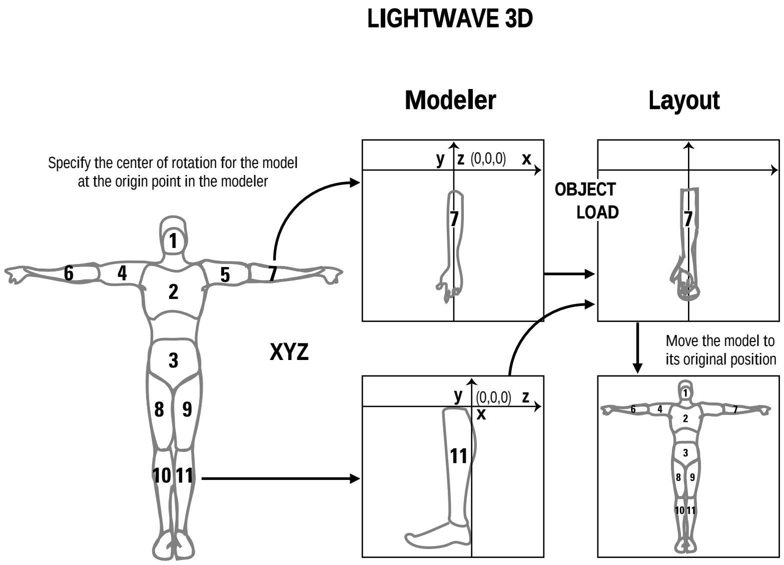
Path Input

If multiple files are found by the automatic path search function, select one file path and input the number. Input the full file path if it is required, then the process is halted by inputting the wrong path three times.

Creating Models for the Converter

The method for creating a parent-child hierarchy is described below.

1) Start up LightWave Modeler.
2) Create the parts and set the materials. Specify the center of rotation of each part as its origin point.
3) One object (lwo file) has now been created.
4) Repeat steps 1 through 3 until all parts have been created.
5) Start up LightWave Layout and load each lwo file.
6) Place each part in the hierarchy and apply the textures.
7) Save the scene files and the object file, convert the model and then display it with the viewer, and confirm that none of the parts are placed in the wrong position in the hierarchy
8) Lastly, apply motion to the object.



Specify the center of rotation for the model at the origin point in the modeler

Move the model to its original position

Textures

The input file formats supported are TGA and BMP.

The texture must be square or rectangle and the size is:

length(1,2,4,8,16,32,64,128,256,512,1024) pixel

width (1,2,4,8,16,32,64,128,256,512,1024) pixel

The converter checks the alpha value of the original image and automatically outputs one of the following three formats accordingly.

No alpha value	: Output in RGB565 format.
Alpha value	: Output in ARGB4444 format.
Alpha value is 0 or 255	: Output in ARGB1555 format.

When the texture is square, the converter automatically selects the twiddled format; when the texture is rectangular, the converter automatically selects the rectangle format.

< twiddled format >

In this texture format, the pixels are re-arranged so that they can be read from memory quickly; this format is supports mip-map; display is faster when this format is used.

<rectangle format>

In this texture format, the pixels are in the same order as that of the original image. Display with this format is slower than with the twiddled format. Note that this format does not support mip-maps.

Texture Extraction (Alpha)

Texture extraction is implemented by three methods in these converters.

1) Extraction by 32-bit TGA

If a 32-bit (ARGB) file is applied to a surface color texture, it is converted as a PVR with alpha. However, because the handling of the alpha portion is different for LightWave and PVR, the image rendered by LightWave will differ from the Ninja display.

2) Extraction by transparency

With this method, the image rendered by LightWave is identical to the Ninja display.

A texture that is a BMP, a 24-bit texture (RGB), or some other texture that does not have an alpha specification is applied to a surface color texture, and the transparency is specified. A 50% specification produces a semitransparent image, and a 100% specification produces a fully transparent image.

3) Extraction by 24-bit BMP + 24-bit BMP

With this method, the image rendered by LightWave is identical to the Ninja display.

First, a 24-bit (RGB) file is applied to a surface color texture. Confirm that the transparency is 0%.

A 24-bit BMP file is then applied to the transparency texture as mask data for extraction. The BMP file that is used for extraction should be 24-bit grayscale. As a result, the converter examines only the red component (8 bits) of the BMP file that is used for extraction, and converts it as the alpha of the PVR file.

The alpha can be extracted with white data. Negative images are not supported.

Material Names

Ninja attributes can be set by material names (material name is the surface name of Light Wave). These attributes are listed below.

It is currently possible to make settings only for material units by this method.

<flags>

D : Set to double-sided polygon.

E : Use environment mapping.

F : Set to flat shading.

L : Ignore light.

<Filter mode>

An : Turn on the anisotropic filter.

Ps : Filter-mode Point Sampled.

Bi : Filter-mode Bilinear filter.

Ti : Filter mode Triliner filter.

D000 : Mip-map `D` adjustment. Adjustment is made by multiplying the mip-map switching Z value of the hardware by this co-efficient. `100` represents a co-efficient of 1.00, and `125` represents a co-efficient of 1.25. This value can be specified in steps of 025 over a range from 025 to 375. If omitted, `D100` is set.

A00 : Alpha blending parameter

Blending is applied as determined by the following equation. The first digit after `A` specifies the field value for `src` and the second digit specifies the field value for `dst`.

```
DTS             : SRC * BlendFunction (SRC Alpha Instruction) +
                  DST * BlendFunction (DST Alpha Instruction)
```

Instruction	Field Value	Values Returned
Zero	0	(0, 0, 0, 0)
One	1	(1, 1, 1, 1)
‘Other’ Color	2	(OR, OG, OB, OA)
Inverse ‘Other’ Color	3	(1 - OR, 1- OG, 1 - OB, 1 - OA)
SRC Alpha	4	(SA, SA, SA, SA)
Inverse SRC Alpha	5	(1- SA, 1 - SA, 1 - SA, 1 - SA)
DST Alpha	6	(DA, DA, DA, DA)
Inverse DST Alpha	7	(1- DA, 1 - DA, 1- DA, 1 - DA)

The default blending parameter is as follows:

Source : SRC Alpha(4)

Destination : Inverse SRC Alpha(5)

Further evaluation of attributes is halted when the underbar (_) character is detected.

Example

For mip-map D adjust 1.25, triliner filter and flat shading:

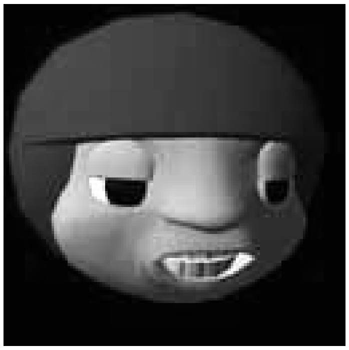
```
D125TiF_mat00
```

Motion Output Using an MRS File

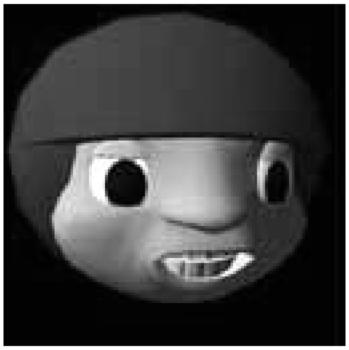
If a motion resource file (.mrs) is used when the basic posture has been determined and then several different motions from that posture are generated, it is possible to eliminate any data for objects, etc., that do not move from the basic posture, thus reducing the amount of data.

Creating Shape Data

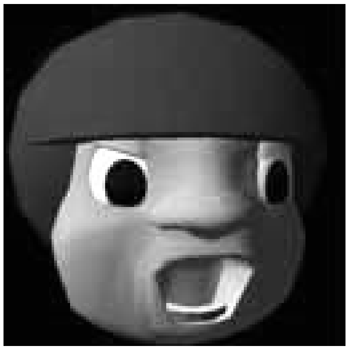
A “shape” is really a motion that changes the shape of a model due to the movement of its vertices. In LightWave, this function is equivalent to the metamorph function, and the metamorph function is used to create shape data. Details on the method of creating shape data in LightWave are provided below.



Object A



Object B



Object C

1) For this example, consider the process of morphing from object A to object B to object C.
2) Create objects A, B, and C.
3) On the Object Panel, set object A as the current object.
4) Set object B as the `Morph Target` object.
5) Click the `E` (envelope) button for the MorphAmount, and set the morph envelope.
6) Create the key frame for starting the morph process, and set the value to 0.
7) Create the key frame for ending the morph process, and set the value to 100.
8) For the transition from B to C, repeat steps (3) through (7) in a similar fashion.

Restrictions

- The number of vertex of each polygon must be 64 or under.
- .mrs, .nja, .nam .pvr, .nas, and .nsc files are created for one hierarchical model.
- The following types of texture mapping are supported : planar image mapping, cylindrical image mapping, and spherical image mapping.
- Use normal-width alphanumerics for texture file names.
- The sequence of rotation in LightWave is Y axis, X axis, Z axis.
- The special plug-in data is not supported basically.

04. Gigen Tool

Function Overview

This command assigns a sequence number to all the .pvr files (PowerVR texture file) in the designated directory and rewrites the globalIndex in the .pvr files. A GBIX chunk will be added to any .pvr file that does not have one.

Parameters

gigen <target directory name> [option] Execution Method

A directory name must include its drive name.

Examples:

```
gigen            c:\3dsoft\texture\
gigen   .        : Specifies the current directory.
gigen   .\test\  : Specifies “test” directory in the current directory.
```

Options

-b baseNo

The starting number (baseNo) of the globalIndex is set by this option.

When this option is not specified, the base number is read from the environmental variable GIGEN_BASE_NO.

-l

Displays file list (being executed)

When a command is being executed, the number assigned to each file and the file name are displayed by this option.

-i

Displays file list only

By specifying this option, the global Index is not rewritten, and the number assigned to each .pvr file and its file name is displayed.

This option can not be used with other options.

05. Ninja Export for 3D Studio MAX R2

Overview

⚠ **NOTE:** This plug-in is intended for 3DStudio MAX R2. Installing this plug-in will render earlier versions (R1, R1.1, and R1.2) unusable.

Ninja Export for 3D Studio MAX R2 is a file export plug-in for Ninja Export for 3DStudio MAX R2 (`Ninja Export`). Specifically, this plug-in can output the following files:

Files used in libraries

Scene files	(.nsc)
Model files	(.nja)
Texture files	(.pvr)
Motion files	(.nam)
Shape motion files	(.nas)

Other files

Motion resource files	(.mrs)
Export log files	(.log)

Installation Method

Copy NJAexp.dle from the Katana release into the `plugins` folder located in the `3dsmax` directory on your hard drive.

Usage

1) After installation, start up `3dsmax`.
2) Create a model, and select `File` from the menu bar, and then select `Save`.
3) When the file browser window appears, input the file name, and select `Ninja FILES (.NSC)` as the file type.
4) When you click the `OK` button, the options dialog box appears.
5) Set the various options, and then click the `OK` button. File output is now complete.

Options Dialog Box

NSC File (Scene)

Create NSC File

If this box is checked, a scene file is output.

NJA File (Model)

Strip

If this box is checked, the mesh is output in strip format.

Add Polygon Normal List

If this box is checked, polygon normal lines are output in an NJA file.

MRS File (Motion Resource)

Create MRS File

If this box is checked, a motion resource list (.mrs) is output.

PVR File (Texture)

Overwrite Texture File

If this box is checked, any texture with the same name in the output destination is overwritten. Normally, no conversion occurs if a file with the same name exists in the output destination.

Texture Global Index Base

The starting number for the globalIndex for textures can be set by inputting a number in this field. The default value is 0.

Texture Type

This item sets the texture type.

Automatic	Switches between Twiddled format and Rectangle format automatically.
Twiddled	All textures are output in Twiddled format.
Rectangular	All textures are output in Rectangular format.

Pixel Format

This option sets the texture output format.

Automatic	Selects the format automatically.
RGB565	Outputs all textures in RGB565 format.
ARGB4444	Outputs all textures in ARGB4444 format.
ARGB1555	Outputs all textures in ARGB1555 format.
Translucent ARGB1555	Outputs all textures in Translucent ARGB1555 format.
Bilevel Translucent ARGB4444	Outputs all textures in Bilevel Translucent ARGB4444 format.

Mipmap OFF

If this box is checked, Mipmap data is not output.

Dither OFF

If this box is checked, texture dithering is not performed.

Alpha Dither OFF

If this box is checked, texture alpha dithering is not performed.

NAM File (TRS Motion)

MRS File

To output motion in terms of the difference from the MRS file, input the name of the MRS file that is to be used in this field. If an MRS file is input in this field, the NSC, NJA, MRS, and PVR items become invisible and only motion-related files are output.

If no MRS file name is input in this field, all files for which output is specified are output.

A detailed description of MRS files follows later in this document.

Motion Type

NoMotion

No motion file (.nam) output.

Type A (TRRR)

Type A motion file output.

In this output format type, `trans` data is output only for the primary node; only `rot` data is output for all other nodes.

Type B (TRTR)

Type B motion file output

In this output format type, `trans` data and `rot` data is output for all nodes.

Type C (TRS)

Type C motion file output

In this output format type, `trans` data, `rot` data, and `scale` data is output for all nodes.

SelectKeys

Ignore Keys (AllFrames)

This option gets data for all frames, generates the key frames on the converter side with the `OptimizeDataSize` option, and then outputs the files.

Key Frames

This option gets data from the positions that have been set as key frames through MAX, and then outputs the files.

OptimizeDataSize

[Key Frame]

Minimize Scale Data

This option cuts those frames for which the amount of change between frames in the scale component of motion is constant.

If this box is not checked, all frames are output, regardless of the amount of change between frames.

Minimize Translation Data

This option cuts those frames for which the amount of change between frames in the translation component of motion is constant.

If this box is not checked, all frames are output, regardless of the amount of change between frames.

Minimize Rotation Data

This option cuts those frames for which the amount of change between frames in the rotation component of motion is constant.

If this box is not checked, all frames are output, regardless of the amount of change between frames.

[Options]

Minimize Scale List

If the scale values and motion data in each `OBJECT` in an `OBJECT` tree are the same and constant, the scale motion for that node is made `NULL`.

If this box is not checked, all data is output, regardless of whether it is the same or not.

Minimize Translation Data

If the translation values and motion data in each `OBJECT` in an `OBJECT` tree are the same and constant, the scale motion for that node is made `NULL`.

If this box is not checked, all data is output, regardless of whether it is the same or not.

Minimize Rotation Data

If the rotation values and motion data in each OBJECT in an OBJECT tree are the same and constant, the scale motion for that node is made NULL.

If this box is not checked, all data is output, regardless of whether it is the same or not.

NAS File (Shape Motion File)

DataType

Don't Create NAS File

No shape motion file (NAS) is output.

Vertex Only

Outputs files that contain vertex information only.

Vertex and Normal

Outputs files that contain both vertex and normal line information.

Select Keys

(Details and cautions concerning shape motion files are provided elsewhere.)

User Keys (Plug-in)

If the KeyListGen plug-in is used in the track view, this option gets the information for the key frames that were created in the `User Keys for Shape` track that was generated.

Modifier Keys

This option gets the key frames from the modifier keys.

Ignore Keys (AllFrames)

This option outputs the data for all frames.

Global Options

XAxis -90 Rotate

This option outputs data for objects and motions with a –90 degree rotation around the X axis in order to bring the MAX coordinate system and the NINJA coordinate system into agreement.

Create Log File

This option outputs various information to a log file during the conversion process.

The file is output with the `.log` extension.

Reset

Clicking this button resets all of the parameters in the dialog box.

Cancel

This button exits without performing the export process.

OK

This button initiates the export process.

Models

This converter outputs one node as a single model file.

If there are multiple nodes, the converter outputs multiple model files.

File names consist of the file name that was input in the export dialog box + an underbar character + the name of the primary object + an extension.

Example: If the file name was test and the object name was `box01`, the name of the model file is `test_box01.nja`.

Never use the same object name twice in one scene. An error will result.

Because the light and camera are output as `NULL` objects during conversion, such objects should be eliminated as much as possible from model output.

Never incorporate a space warp into a model node.

Outputting Selected Models

If objects are selected within a scene, only the NJA files for the nodes to which the selected objects belong are output.

At present, up to 100 nodes can be selected. As long as the number of selected nodes is under that limit, any number of objects can be selected. (If too many are selected, efficiency will suffer. Refer to the explanation below.)

When outputting selected models, efficiency is better when only a few portions of a model are selected. For example, when outputting one of five models, processing is faster if only one part of the model (a hand, for example) is selected instead of the entire model. The reason why is because for every part that is selected, a check is made of which model each part belongs to, and duplicate models are rejected.

Materials

Only “Standard Materials” and “Multi/Sub-Object Materials” are supported. Do not set any other type of material.

Do not create more than one layer of “Multi/Sub-Object Materials”.

There must only be one layer of “Multi/Sub-Object Materials”. Each sub-material is handled in the same manner as a “Standard material”.

Within the “Standard Material” rollout, the parameters that are used are mirrored light, diffuse reflective light, and opacity.

If no material is assigned to an object, the wire color (the color of the object) is output as the material.

Textures

The following file formats can be used for textures: BMP and TGA.

Texture sizes can range from 1, 2, 4, 8 16, 32, 64, 128, 256, 512, or 1024 pixels in the vertical direction by 1, 2, 4, 8 16, 32, 64, 128, 256, 512, or 1024 pixels in the horizontal direction, and can be either square or rectangular.

At present, there are no restrictions concerning the directory in which textures can be stored.

Conversion from multiple directories is supported.

When using textures, be certain to define the materials first before executing diffuse reflective light mapping.

Textures other than diffuse reflective light and opacity are not supported.

Diffuse reflective light and opacity are the only textures that can be defined for one (sub-)material.

When a texture has been defined, be certain to load the bitmap data (.bmp, .tga) for the bitmap parameters.

The only data that is used in each rollout (coordinates, noise, bitmap parameters, etc.) for diffuse reflective light and opacity maps is the name of the bitmap file. Any other data that is set is ignored.

If `Automatic` is selected for each option, the converter checks the alpha value of the original graphic and outputs the texture in one of the following three formats automatically.

If there is no alpha value : The converter outputs the texture in RGB565 format.

If there is an alpha value : The converter outputs the texture in ARGB4444 format.

If the alpha value is either 0 or 255 : The converter outputs the texture in ARGB1555 format.

In addition, if the texture is square, the converter automatically selects twiddled format; if the texture is rectangular, the converter automatically selects rectangle format.

<twiddled format>

In a texture in this format, the pixels have been rearranged into an order that permits them to be read out of memory quickly. This format permits use of Mipmap textures. Display is fast with this format.

<rectangle format>

In a texture in this format, the order of the pixels is the same as in the image. Display is slower than with twiddled format. Note that Mipmap textures cannot be used with this format.

Texture Extraction (Alpha)

Texture extraction is implemented by two methods in this converter.

Method 1: Extraction by 32-bit TGA

If a 32-bit (ARGB) file is applied to a diffuse reflective light map, it is converted as PVR with alpha. However, because the handling of the alpha portion is different between MAX and PVR, the image rendered by MAX will differ from the Ninja display.

In the case of this method (tga), a texture that is applied to an opacity map is ignored.

Method 2: Extraction by 24-bit BMP + 24-bit BMP

With this method, the image rendered by MAX is similar to that of the Ninja display.

First, a 24-bit (RGB) file is applied to a diffuse light map.

A 24-bit BMP file is then applied to the opacity map as mask data for extraction. The BMP file that is used for extraction should be 24-bit grayscale. As a result, the converter examines only the red component (8 bits) of the BMP file that is used for extraction, and converts it as the alpha of the PVR file.

If a BMP file is not applied to the diffuse reflective light map, the opacity map is ignored.

Texture UV

In MAX, the UV (repeat, etc.) settings can be made on the object (mesh) side and the material (texture) side.

In the current version (Rel 0.72a1), the model side supports UV settings made through the UVW map modifier (method of application, repeat, etc.) and the material side supports Tiling and Mirroring (flip)

These functions can be combined.

Basically, if a method corresponding to that described above is used, the image rendered by MAX and the image output by NINJA will match.

Be certain that either the `tiling` checkbox or the `mirror` checkbox in the coordinate rollout on the material (map) side is checked. If neither of these is checked, the rendered image and the output results may differ. Basically, because the `tiling` checkbox is checked by default, no special attention is required regarding this item.

In addition, when a BMP file is applied to opacity, the MAX and Ninja images can be made to match by setting the tiling value or mirror value to the same values that were set for diffuse reflective light.

TRS Motion

`TRS motion` is translation, rotation, and scale motion for individual objects.

With this converter, data can be retrieved by the following two methods:

Ignore Keys (All Frames)

This method outputs TRS motion for all frames.

When this option is set to its default setting, optimization (key generation) is applied internally. If these options are turned off, detailed motion can be output.

Key Frames

This option gets data from the key positions for TRS that were set through MAX.

This method outputs data efficiently, however, an error may result if an attempt is made to use this data output type with special objects (such as bones).

Bone-based motion can also be used.

In this case, the bones themselves are converted into `NULL` objects. An object that includes bones should be output using `Ignore Keys.`

⚠ **NOTE:** Do not use negative frames.

Motion Output Using an MRS File

If a motion resource file (.mrs) is used when the basic posture has been determined and then several different motions from that posture are generated, it is possible to eliminate any data for objects, etc., that do not move from the basic posture, thus reducing the amount of data.

For details, refer to the document *"Reminders of Ninja Motion"*.

Shape Motion

There are three methods (described below) for obtaining vertex animation. These methods can be specified in the export option dialog box.

UserKeys (Plugin)

With this method, data can be gotten from frames specified by the user by using the user key generation plug-in (KeyListGen) provided.

Note that because the key positions specified by this method are specified globally, when a frame is output with an object in a node the same shape key is set for all objects.

Therefore, vertex data is output for some models even if they have no shape motion.

Modifier Keys

This method gets key frames from the key positions indicated by the modifier parameters.

Although separate keys can be obtained for each object, the number of keys becomes large if a large number of modifiers is used or if a complex deformation with a large number of keys is added.

Note also that some keys for some parameters may not be supported.

Ignore Keys (All Frames)

This method gets vertex data from all frames.

An extremely large amount of data is generated if a large number of vertices and frames are involved.

Attributes

Double-sided Polygons

The double-sided polygon attribute can be set for individual polygons according to the material.

This attribute is defined by checking the double-sided checkbox in the material rollout.

Flat Shading

Although gouraud shading is normally used for all models, flat shading can be specified for individual polygons according to the material, when desired.

This attribute is defined by setting `Shading` to `Flat` in the material rollout.

Attributes That Are Set by Adding Characters at the Beginning of Material Names

The following attributes can be set by changing the names of materials.

<flags>

E : Environment Mapping

To set environment mapping, apply a texture to a diffuse reflective light map normally, and then add `E` at the beginning of the material name.

L : Ignore light source calculation

This attribute can be defined by adding an `L` at the beginning of the material name for a model or polygon for which lighting should be ignored.

<Filter mode>

An	Turns the anisotropic filter on.
Ps	Filter-mode Point Sampled.
Bi	Filter-mode Bilinear Filter.
Ti	Filter-mode Triliner Filter.

Blend Texture is set automatically when there are two textures.

D000 : Mipmap `D` adjustment. Adjustment is made by multiplying the hardware's own mipmap switching Z value by this coefficient. The value `100` means 1.00 and `125` means 1.25. This parameter can be set in a range from 025 to 375, in steps of 025. If omitted, `D100` is set.

A00	Alpha blending parameter

The blending that is performed is determined by the following formula. The src Filed Value is specified by the first digit after `A`, and the dst Filed Value is specified by the second digit after A.

```
DST := SRC * BlendFunction(SRC Alpha Instruction) +
DST * BlendFunction(DST Alpha Instruction)
```

Instruction	Field Value	Values Returned
Zero	0	(0, 0, 0, 0)
One	1	(1, 1, 1, 1)
'Other' Color	2	(OR, OG, OB, OA)
Inverse 'Other' Color	3	(1 - OR, 1- OG, 1 - OB, 1 - OA)
SRC Alpha	4	(SA, SA, SA, SA)
Inverse SRC Alpha	5	(1- SA, 1 - SA, 1 - SA, 1 - SA)
DST Alpha	6	(DA, DA, DA, DA)
Inverse DST Alpha	7	(1- DA, 1 - DA, 1- DA, 1 - DA)

If the underbar character (_) is detected, subsequent parameters are not evaluated.

Example

To specify Mipmap D adjust 1.25, triliner filter, and flat shading:

```
D125TiF_mat00
```

Options that are specified by adding an uppercase letter at the beginning of a material name can be specified in any sequence, as long as they are at the beginning of the name. However, if a material name begins with one of these letters, the name must be changed so that it begins with a different letter in order to prevent the corresponding attribute from being set inadvertently.

Because the default material name in 3DStudioMAX begins with `Material`, the character string `Material` is not deemed to begin with the `Ma` flag. to set the `Ma` flag, add `Ma` in front of `Material`.

Error Messages

The error messages that are output in the dialog boxes are listed below.

Error message	Meaning
Too many objects are selected!!	Too many objects were selected.
Fail to Create LOG file.	Attempt to create a LOG file failed.
There is No User Key List.	No UserKey has been set.
No source data!!	There is no model in the scene.
Bad nodelist	Illegal node list.
Fail to read MRS file.	Attempt to read an MRS file failed.
Fail to Create MRS file.	Attempt to create an MRS file failed.
Fail to Create NJA file.	Attempt to create an NJA file failed.
Fail to Create NSC file.	Attempt to create an NSC file failed.
Fail to Create NAS file.	Attempt to create an NAS file failed.
Fail to Create NAM file.	Attempt to create an NAM file failed.
Finished!!	Output was completed successfully.

Other Notes

Always be sure to save files in max format before outputting them.

This plug-in does not support special plug-in data.

Data that is retrieved and data that is output by this plug-in conforms with the Ninja specifications.

Future Development

Support for scenes (cameras, lighting)

We intend to make it possible to permit the setting of UserKey for individual objects.

Bug fixes.

06. KeyListGen (3D Studio Max R2)

Overview

⚠ **Note:** This plug-in is intended for 3DStudio Max R2. Installing this plug-in will render earlier versions (R1, R1.1, and R1.2) unusable.

KeyListGen is a user key generation plug-in for "Ninja Export for 3DStudio Max R2" (Ninja Export).

Currently, this plug-in can be used to retrieve data from any frame set by the user for (shape) motion in which vertices move.

Usage

For details on the installation procedure, refer to the Ninja Export documentation.

Open the track view window, click on the `Track View Utility` icon in the upper right corner, and select `KeyListGenerator`.

This operation adds the `UserKeys for Shape` track to the track view.

The user then presses the key for the frame position to be obtained.

If `User Keys (Plugin)` (default) is selected in the Select Keys combo box for `NAS File (Shape Motion)` in the options window when Ninja Export is executed, the data is retrieved from the frame position that was specified by the user.

Restrictions

The key position specified by this plug-in is a global specification. When outputting objects that are in a parent-child hierarchy, the same shape key is set for all of the objects.

Therefore, for some models, vertex data is output even if there is no shape motion.

Future Development

We intend to permit keys to be set for individual objects.

07. NinjaExport for SoftImage3D NT

ninjaExport Overview

- This ninjaExport is for SoftImage3.7.
- It outputs Ninja model format files (.nja) and Ninja motion format (.nam) directly from SoftImage.
- The model hierarchy information file .mrs files can be output simultaneously.
- Output files (.nja), (.nam), (.mrs) are in ASCII code.
- It can be started from the `Export/Object/NinjaExport` menu within the tools mode of SoftImage.

ninjaExport Package Contents

ninjaExport is composed of the following 4 files.

- ninjaExport.dll- Program itself (Dynamic Link Library)
- NinjaExport.cus- Menu Script

ninjaExport Installation method

- Login and work in the super user mode.
- Check the directory in which SoftImage has been installed.
  (Here, it is assumed that SoftImage has been installed in `Soft3d_3.7.`)
- Copy ninjaExport.dll to `...Soft3d_3.7\3D\custom\bin`.
- Copy NinjaExport.cus to `...Soft3d_3.7\3D\custom\tools`.
- Now you can execute ninjaExport.

ninjaExport Functions

Setting for File Output

- The output file path can be specified in the `Output Directory:`
- When `Output Directory:` is not specified, the output files (.nja), (.nam) and (.mrs) are created in the directory where SoftImage was executed.
- Users can select whether outputs the nja file (`Yes`) or not (`NO`) by `Output Modelfile(.nja):`.

  The default of `Output Motionfile(.nja):` is `Yes`.
- Users can select whether outputs the nam file (`Yes`) or not (`NO`) by `Output Modelfile(.nam):`.

  The default of `Output Motionfile(.nam):` is `Yes`.
- Users can select whether outputs the mrs file (`Yes`) or not (`NO`) by `Output Mrsfile:`.

  The default of `Output Mrsfile:` is `Yes`.

Options for Models

- Users can specify whether add the polygon normal vector to the nja file or not by `Use Polygon Normal List:`.
- The default of `Use Polygon Normal List:` is `No`.

Texture-related Options

- Users can specify the base number of GlobalIndex for textures by `GlobalIndexNo of Textures:`
- The default of `GlobalIndexNo of Textures:` is `0`.
- Users can specify whether overwrite files or not by `Force texture covert`, even if the texture file already exists. By selecting this option, files are overwritten.
- Texture type can be specified by `Texture Type:`.

The types are:

Auto / Twiddled / Rectangle

- `Twiddled` and `Rectangle` can be switched automatically by `Auto`.
- The output format of textures can be set by `Pix Format:`.

The formats are:

Auto / RGB565 / ARGB4444 / ARGB1555 / Translucent ARGB1555 / Bilevel Translucent ARGB4444

- Formats are selected automatically by `Auto`.
- Mipmap switch : controls the mipmap.
- Mipmap switch : has three items of `Auto Mipmap Off` / `Dither Off` / `Alpha Dither Off`.
- The automatic Mipmap generation is halted by selecting (Auto Mipmap Off).
- (Dither Off : turn the Dither OFF).
- (Alpha Dither Off : turn the Alpha Dither OFF).

Motion-related Options

- Users can specify motion types by `MotionType:`
- The default of `MotionType:` is `TypeA(TRRR)`.
- `TypeA(TRRR)` expresses motions using `Translation` and `Rotation` for the root node, and using `Rotation` for the node excepting root.
- `TypeB(TRTR)` expresses motions using `Translation` and `Rotation` for all nodes.
- `TypeC(TRS)` expresses motions using `Translation`, `Rotation` and `Scaling` for all nodes.
- `Print all scale array` is the item to output all scale values without optimization (deletion) even if the same scale values continue.
- `Print all translation array` is the item to output all translation values without optimization (deletion) even if the same coordinates continue.
- `Print all rotation array` is the item to output all rotation values without optimization (deletion) even if the same rotation values continue. If this item is selected, all rotation values are output.
- `Print all scale keyframe` is the item to output all frames between scale keyframes.

  If this item is selected, all frames are output.
- `Print all translation keyframe` is the item to output all frames between coordinate keyframes.

  If this item is selected, all frames are output.
- `Print all rotation keyframe` is the item to output all frames between rotation value keyframes.

  If this item is selected, all frames are output.

ninjaExport Texture

The overview of the textures is as follows…

The converter checks the alpha value of the original image and outputs textures of the following three formats.

RGB565 : All alpha values are 255

ARGB1555 : Alpha values are 0 or 255

ARGB4444 : Alpha values change between 0-255

Also, in the case that the texture is square, the texture format becomes `twiddled` automatically, and in the case that the texture is rectangle, the format becomes `rectangle` automatically.

twiddled format

`twiddled format` texture is the one which rearranges pixels of the texture to be read at high speed. Mipmap is available and display is at high speed in this format.

rectangle format

`rectangle format` texture is the one which does not rearrange pixels and leave it as it is.

Lower speed than `twiddled format`. Note that Mipmap is not available.

ninjaExport Material

The following attributes can be specified for the material names.

In future versions, we plan to add support for settings by polygon unit in addition to material names.

flags

D : Double-sided polygon

E : Environment mapping

F : Flat shading

L : Ignores Light

Filter mode

An : anisotropic filter ON

Ps : Filter-mode Point Sampled

Bi : Filter-mode Bilinear Filter

Ti : Filter-mode Triliner Filter

D000 : Adjustment of Mip-map `D` adjust. Adjusts it by multiplying the switch z value of the hardware mipmaps by this coefficient. 100 means 1.00 and 125 means 1.25. The values from 025 to 375 can be specified by the 025 unit. The default value is D100.

A00 : Alpha blending parameter

Blend is expressed by the following equations. The first numeral after A specifies the filed value of `src` and the second numeral after A specifies the filed value of `dst`.

```
DST := SRC * BlendFunction(SRC Alpha Instruction) +
       DST * BlendFunction(DST Alpha Instruction)
```

Instruction	Field Value	Values Returned
Zero	0	(0, 0, 0, 0)
One	1	(1, 1, 1, 1)
'Other' Color	2	(OR, OG, OB, OA)
Inverse Other' Color	3	(1 - OR, 1- OG, 1 - OB, 1 - OA)
SRC Alpha	4	(SA, SA, SA, SA)
Inverse SRC Alpha	5	(1- SA, 1 - SA, 1 - SA, 1 - SA)
DST Alpha	6	(DA, DA, DA, DA)
Inverse DST Alpha	7	(1- DA, 1 - DA, 1- DA, 1 - DA)

If '_' (under bar) is detected, subsequent parameters are not evaluated.

Examples

In the case of Mip-map D adjust1.25, Triliner Filter, flat shading

```
D125TiF_mat00
```

ninjaExport Future Support Plan

- BIN typed Ninja format files (.njd, .njm,)
- .njd: model files, .njm: motion files

08. NinjaExport for SoftImage SGI

Overview

The following diagram illustrates the flow of data using HrcToNj:

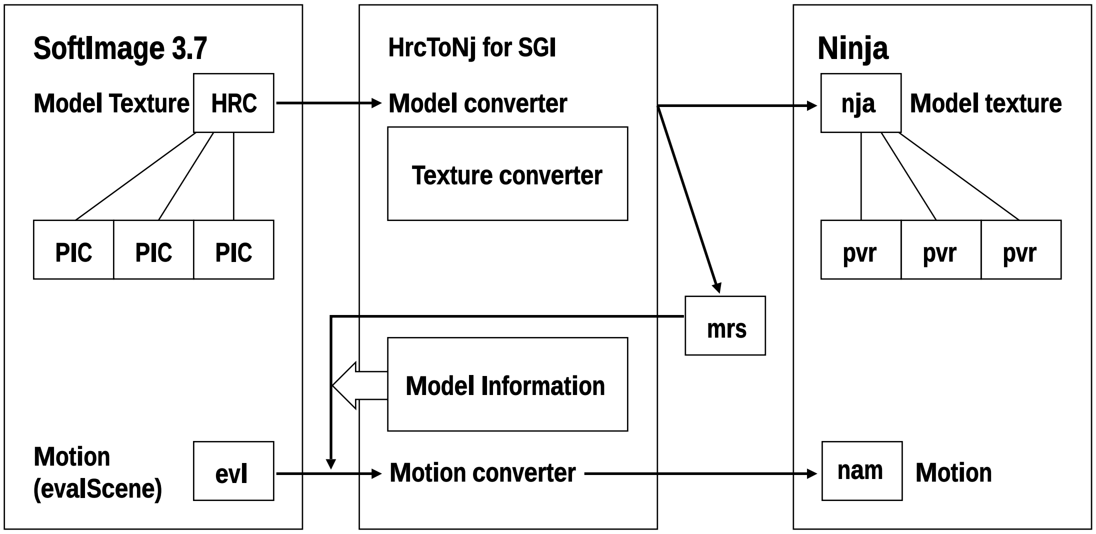


Figure 1

- `HrcToNj` runs on the SGI.
- On the basis of an input `hrc` file, `HrcToNj` outputs an `mrs` file, which contains the model and parent-child hierarchy information. If textures are used in the model in the `hrc` file, the texture converter `pvrconv` is called internally to perform the processing (converting `pic` to `pvr` with a `mipmap`).
- `HrcToNj` generates the Ninja motion `nam` file from the `mrs` file and the `evalScene` file.
- `HrcToNj` adds a `globalIndex` to the textures. Under the default setting, numbers are assigned to all textures, starting from `0`. The `globalIndex` numbers are used to detect duplicate textures among multiple models

- gigen is used in order to assign unique globalIndex numbers among multiple model (nja) files.
- gigen creates a pvr list from the multiple nja and pvr files in the specified directory, generates unique globalIndex numbers, and assigns the globalIndex numbers to all of the nja and pvr files.
- The base number of the globalIndex can be specified by using the -g option for HrcToNj.

Flow for the gigen (globalIndex Generator) Command

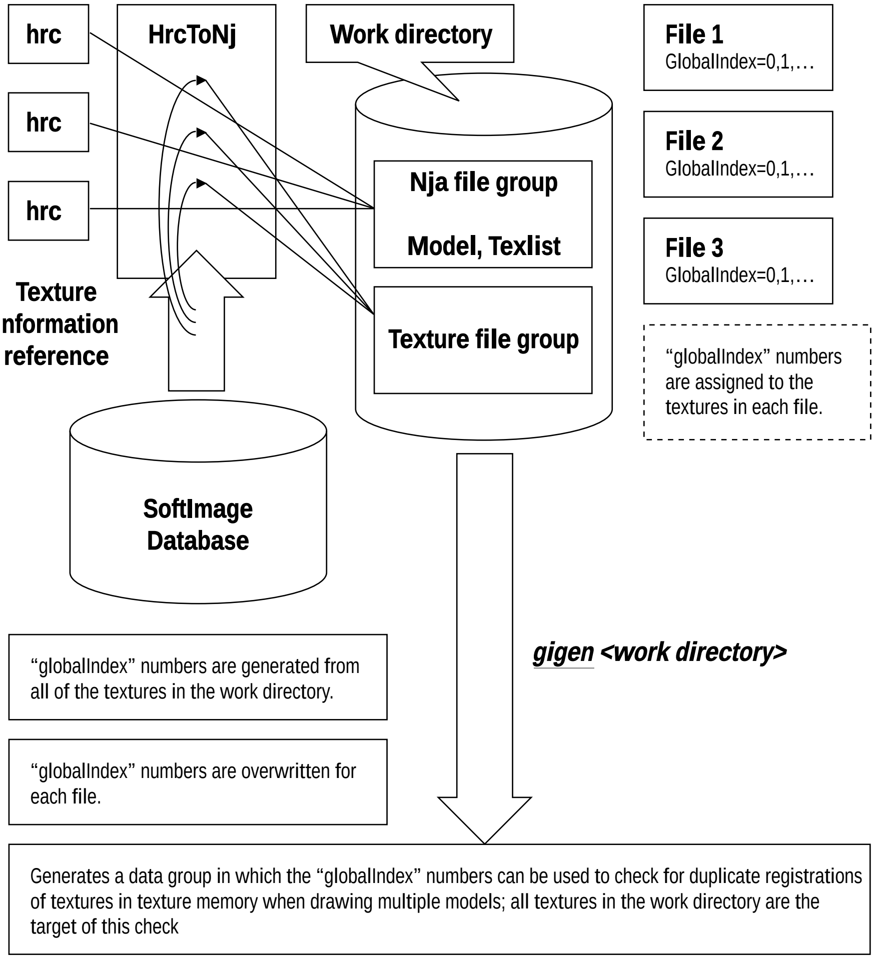

hrc
hrc
hrc
HrcToNj
Work directory
Nja file group
Model, Texlist
Texture file group
File 1
GlobalIndex=0,1,…
File 2
GlobalIndex=0,1,…
File 3
GlobalIndex=0,1,…
"globalIndex" numbers are assigned to the textures in each file.
Texture information reference
SoftImage Database
gigen <work directory>
"globalIndex" numbers are generated from all of the textures in the work directory.
"globalIndex" numbers are overwritten for each file.
Generates a data group in which the "globalIndex" numbers can be used to check for duplicate registrations of textures in texture memory when drawing multiple models; all textures in the work directory are the target of this check

Figure 2

Because gigen permits the user to set the base value for globalIndex, assigning numbers that are widely separated between individual groups makes it possible to execute gigen just for specific group units.

Example:

```
gigen  /tmp/data1 –b 10000
gigen  /tmp/data2 –b 20000
```

To check for duplicates among all of the data, it is necessary to execute "gigen" once for all of the data at one time.

Notes on the Ninja File Format and Compiling

The following data is output.

.nja: Model data, texture list (texlist)

.nam: Motion data

These are generated in ASCII for format for compiling, and have a format that can be replaced with a C description through a `define` statement. When compiling this data, it is necessary to declare `NjDef.h` in an `include` statement.

Example:

```
Test.c
--
#include “Ninja.h”
#include “NjGlobal.h”
#include “NjDef.h”    /*  Place before "nja" and "nam".  (Important!!) */
#include “test.nja”
#include “testmot.nam”

main()
{
}
--
```