

N I N T E N D O
NITRO-System
NITRO Intermediate File Format
Version 1.6.0

The contents in this document are highly
confidential and should be handled accordingly.

"Confidential"

These coded instructions, statements, and computer programs contain proprietary information of Nintendo of America Inc. and/or Nintendo Company Ltd., and are protected by Federal copyright law. They may not be disclosed to third parties or copied or duplicated in any form, in whole or in part, without the prior written consent of Nintendo.

© 2004-2005 Nintendo

TM and ® are trademarks of Nintendo.

Dolby, Pro Logic and the Double-D symbol are trademarks of Dolby Laboratories.

IBM is a trademark of International Business Machines Corporation.

Roland GS Sound Set is a trademark of Roland Corporation U.S.

All other trademarks and copyrights are property of their respective owners.

Confidential

These coded instructions, statements, and computer programs contain proprietary information of Nintendo of America Inc. and/or Nintendo Company Ltd. and are protected by Federal copyright law. They may not be disclosed to third parties or copied or duplicated in any form, in whole or in part, without the prior written consent of Nintendo.

Table of Contents

1	Introduction.....	1
2	Intermediate Files.....	3
2.1	What Are Intermediate Files?	3
2.1.1	Types of Intermediate Files.....	3
3	Intermediate File Description Rules.....	5
3.1	The Structure of Intermediate Files	5
3.1.1	XML Declaration	5
3.1.2	The Structure of Elements in the Intermediate File.....	5
3.2	Elements.....	6
3.2.1	Naming Rules for Elements	6
3.2.2	Other Rules	6
3.3	Attributes.....	7
3.3.1	Naming Rules for Attributes	7
3.3.2	Other Rules	7
3.4	Explanation of Common Header Elements and Attributes	8
3.4.1	The Root Element	8
3.4.2	The <head> Element	8
3.4.3	The <create> Element	8
3.4.4	The <title> Element.....	9
3.4.5	The <comment> Element.....	9
3.4.6	The <generator> Element	10
3.4.7	The <body> Element.....	10
3.5	Other Rules.....	11
4	Intermediate File for Model Data	13
4.1	Model Data Files (imd).....	13
4.1.1	Overview of imd Files.....	13
4.1.2	The Basic Element Structure of imd Files.....	14
4.1.3	Description of imd Elements and Attributes	15
4.1.4	Supplemental Information for imd Files.....	48
5	Intermediate Files for Animation.....	55
5.1	The Data Formats of Animation Files	55
5.1.1	Types of Data Formats	55
5.1.2	Frame Format	55
5.1.3	FV Key Format.....	58
5.2	Character Animation Files (ica)	60
5.2.1	Overview of ica Files.....	60
5.2.2	The Basic Element Structure of ica Files	60
5.2.3	Explanation of ica Elements and Attributes.....	60
5.2.4	Supplemental Information for ica Files.....	67
5.3	Visibility Animation Files (iva)	67
5.3.1	Overview of iva Files.....	67
5.3.2	The Basic Element Structure of iva Files	67
5.3.3	Explanation of iva Elements and Attributes.....	67
5.3.4	Supplemental Information for iva Files.....	71
5.4	Material Color Animation Files (ima).....	72
5.4.1	Overview of ima Files.....	72
5.4.2	The Basic Element Structure of ima Files.....	72
5.4.3	Explanation of ima Elements and Attributes	72
5.4.4	Supplemental Information for ima Files.....	80

5.5	Texture Pattern Animation Files (itp)	81
5.5.1	Overview of itp Files.....	81
5.5.2	The Basic Element Structure of itp Files.....	81
5.5.3	Explanation of itp Elements and Attributes	81
5.5.4	Supplemental Information for itp Files.....	87
5.6	Texture SRT Animation Files (ita).....	87
5.6.1	Overview of ita Files.....	87
5.6.2	The Basic Element Structure of ita Files.....	87
5.6.3	Explanation of ita Elements and Attributes	88
5.6.4	Supplemental Information for ita Files.....	94
6	Intermediate Files for Graphic Scenes	95
6.1	Graphic Scene Data File (isd).....	95
6.1.1	Overview of isd Files.....	95
6.1.2	Specific isd Element Structure	95
6.1.3	isd Elements and Attributes	95
6.1.4	isd Supplement	104
7	Optimizing Data	105
7.1	Culling Nodes	105
7.2	Material Compression	105
7.3	Making Polygon Strips	106
8	Reference Information	107
8.1	Matrix Computations.....	107
8.1.1	Node Matrix Computation Methods.....	107
8.1.2	Texture Matrix Computation Methods	107

Revision History

Version	Revision Date	Description
1.6.0	2/2/2005	<ul style="list-style-type: none"> Added and deleted attributes in imd: <ul style="list-style-type: none"> Added the attribute <code>tex_effect_mtx</code> to the <code><material></code> element (4.1.3.14) Deleted the attribute <code>tex_gen_st</code> from the <code><material></code> element Added imd descriptions: <ul style="list-style-type: none"> Added a description of the attribute <code>tool_start_frame</code> for the <code><model_info></code> element (4.1.3.3) Revisions to and addition of the description of the attributes <code>tex_gen_mode</code> and <code>tex_gen_st_src</code> for the <code><material></code> element (4.1.3.14) Revised the output conditions for the <code><tex></code> element (4.1.3.27) Added the description "about environmental and projection mapping" to the supplemental information for imd files (4.1.4) Revised the description of section supplemental information for isd files (6.1.4)
1.5.0	12/13/2004	<ul style="list-style-type: none"> Added imd element attributes Added <code><original_create></code>, <code><original_generator></code> elements (4.1.3.1, 4.1.3.2) Added attributes <code>depth_test_decals</code>, <code>render_1_pixel</code>, <code>far_clipping</code>, <code>translucent_update_depth</code> to the element <code><material></code> (4.1.3.14) Added the attribute <code>priority</code> to the <code><display></code> element. Wrote a description related to rendering priority (4.1.3.40) Added elements, attributes to isd: <ul style="list-style-type: none"> Added <code><y_sorting></code> element (6.1.3.11) Added <code><render_1_pixel_depth></code> element (6.1.3.15) Added attributes <code>depth_buffer</code>, <code>scale_w</code> to <code><camera></code> element (6.1.3.1) Revised supplemental Information for ica files "Notes on replay with the G3D Library" (5.2.4)
1.4.2	10/25/2004	<ul style="list-style-type: none"> Revised the description of <code>light0</code> in "Table 4-12 The attributes of the <code><material></code> element (4.1.3.14). Revised and supplemented the description of the interpolation attribute in <code>ica</code>, <code>ima</code>, <code>ita</code>
	8/30/2004	<ul style="list-style-type: none"> Changed output conditions for imd <code><node></code> element attribute <code>scale_compensate</code>. Added explanation of "Envelopes in NINTENDO NITRO-System" Added explanation of normal vectors to "The Matrix that Transforms Vertex Position Coordinates".
1.4.1	8/2/2004	<p>Revised to support:</p> <ul style="list-style-type: none"> Nitro Intermediate File Plug-in for SOFTIMAGE XSI Nitro Intermediate File Plug-in for 3ds max Added description of the <code>scaling_rule</code> attribute in the imd, ica <code><*** info></code> element (tables 4-1 and 5-4) Added <code> xsi, 3dsmax</code> to the values taken by the <code>tex_matrix_mode</code> attribute in the imd, ita <code><*** info></code> element (tables 4-1 and 5-36) Revised description of the <code>name</code> attribute in the imd <code><polygon></code> element
1.4.0	6/30/2004	<ul style="list-style-type: none"> Revised description of <code>name</code> attribute for imd <code><polygon></code> element Revised description of <code>twist</code> attribute for isd <code><camera></code> element Made additions to "6.1.4 isd Supplement"

Version	Revision Date	Description
1.3.0	5/24/2004	<ul style="list-style-type: none">• Added scaling rule attribute to ica <node_anm_info> element• Revised the output conditions in the <palette_name> element• Added to a description and diagram to the <camera> element
1.2.0	4/12/2004	<ul style="list-style-type: none">• Added an intermediate file for graphic scenes (isd)• Changed output conditions for the source attribute in The attributes of the <create> element table• Revised the explanation in The <comment> element description• Corrected the value taken by the polygon_mode attribute of the <material> element of imd from toon_hilight to toon_highlight• Added and revised text in conjunction with support for the 3D Material Editor:<ul style="list-style-type: none">• Added explanatory text.• Added elements and attributes to imd that are specific to the 3D Material Editor.
1.1.1	3/25/2004	<ul style="list-style-type: none">• Made additions in accordance with support for the intermediate file output plug-in for SOFTIMAGE 3D.• Added Types of null culling to the Types of Node Culling table.
1.1.0	3/1/2004	<ul style="list-style-type: none">• Added the use_primitive_strip attribute to the imd <model_info> element.• Added support for the A3I5 translucent texture.• Deleted 2 from the value obtained by the color_size attribute of the imd <tex_palette> element.• Changed the limitation on the number of characters usable for names in the manual's Nintendo NITRO-System library from 15 to 16.• Revised the explanatory sentences in the Connecting Polygons section.• Revised the explanatory sentences in the Node Matrix Computation Methods section.
1.0.0	1/23/2004	Initial version.

1 Introduction

The data files used for displaying 3D models on NITRO are known as NITRO Intermediate files. This manual is designed to provide an understanding of NITRO Intermediate files for programmers who intend to create their own converters and display routines.

Because NITRO Intermediate files are XML-format files, knowledge of XML is necessary to correctly understand the material in this manual.

2 Intermediate Files

This chapter describes how to install the 3D Material Editor. It also describes the 3D Material Editor's folder organization.

2.1 What Are Intermediate Files?

Intermediate files are XML-format text files that store 3D model and related animation data in the data structure appropriate for NITRO.

These Intermediate files can be output from your 3D computer graphics tools using the **NITRO Intermediate File Plug-in**. Once you have output Intermediate files, you can edit the material properties using the Nintendo NITRO-System 3D Material Editor (hereafter referred to as 3D Material Editor) and then display the data on NITRO using the Nintendo NITRO-System library. Alternately, you can analyze the contents of the Intermediate files and build your own display routines.

As of February 2005, Intermediate file output is supported for the following 3D computer graphics tools:

- NITRO Intermediate File Plug-In for Maya
- NITRO Intermediate File Plug-In for SOFTIMAGE|3D
- NITRO Intermediate File Plug-In for SOFTIMAGE|XSI
- NITRO Intermediate File Plug-In for ds max

Please refer to the manuals for details on these plug-ins.

Note: SOFTIMAGE|3D is not currently supported for the US market. If you want to develop using SOFTIMAGE|3D, please contact support@noa.com.

This is Version 1.6.0 of the Intermediate File Format.

2.1.1 Types of Intermediate Files

There are seven different kinds of NITRO Intermediate files separated by function that are distinguished by their file extensions.

All of the data necessary for display of 3D models is stored in files with the `.imd` extension. Based on this imd file, you can use various other Intermediate files for animation to do such things as move by changing the pose (character animation), toggle between showing and hiding at the node unit (visibility animation), change the material color (material color animation), substitute textures (texture pattern animation), and scroll the texture (texture SRT animation).

The isd files store information for displaying 3D models to configure scenes in frame units, primarily camera, lighting, and toon tables, that cannot be stored in imd files.

Table 2-1 - Types of Intermediate Files

Extension	Type of Intermediate File	File Contents
imd	Model data	Model information including polygons, parent/child structures, materials and textures.
ica	Character animation data	Animation information for controlling node matrices.
iva	Visibility animation data	Animation information for controlling node visibility.
ima	Material color animation data	Animation information for controlling material color.
itp	Texture pattern animation data	Animation information for replacing multiple textures.
ita	Texture SRT animation data	Animation information for controlling texture matrices.
isd	Graphic scene data	Scene information such as camera, lighting, and toon tables.

Note: Currently, isd files can only be output from the 3D Material Editor.

3 Intermediate File Description Rules

The 3D Material Editor application runs on Windows. This chapter gives an overview of the 3D Material Editor. It also describes its main features and window organization.

3.1 The Structure of Intermediate Files

3.1.1 XML Declaration

To indicate that the file is an XML document, the first line of the file must be the following XML declaration:

```
<?xml version="1.0" encoding="Shift_JIS"?>
```

3.1.2 The Structure of Elements in the Intermediate File

The *root* element is the element for all Intermediate files with the same extension name. Stored under each root element is the header part of the file (the `<head>` element) and the data part of the file (the `<body>` element). The `<head>` element stores information regarding the document itself in a format that is common to all Intermediate files. The element structure of the `<body>` element differs for each type of Intermediate file.

Example of an imd file:

<code><?xml version="1.0" encoding="Shift_JIS"?></code>	... XML declaration
<code><imd version="1.2.0"></code>	... Root element
<code><head></code>	... Header part
... (Omitted: Common to all Intermediate files) ...	
<code></head></code>	
<code><body></code>	... Data part
... (Omitted: Differs depending on Intermediate file type) ...	
<code></body></code>	
<code></imd></code>	

3.2 Elements

3.2.1 Naming Rules for Elements

- Element names are assembled from English words made from lower-case letters and the underscore character (_). The names do not include uppercase letters or Japanese romaji expressions.
- Some element names are abbreviated. The following list is presented alphabetically:

animation	anm
color	clr
difference	diff
index	idx
information	info
matrix	mtx
normal	nrm
position	pos
primitive	prim
scale rotate translate	srt
texture	tex
vertex	vtx

- An element with the name `<XXXX_array>` always has a *size* attribute, and it stores *size* number of `<XXXX>` child elements.
- As a rule, element names are not duplicated among the Intermediate files. In almost all cases, the name of the element determines in which file and where the element is output. As an exception, there are some elements that are shared by the Intermediate files used by animation. These elements use a standard format.

3.2.2 Other Rules

- For Intermediate files, unnecessary elements that do not contain data are not output. Certain elements are only output under specific conditions. To learn about these, see the "Output condition" line in the description of each element below.
- Elements with the *index* attribute are arranged in order of the index.
- In general, fixed-length data are held in attributes. For variable-length data, there is always an attribute that holds a numerical value indicating the size of the data, and the data array is stored as the content of the element.
- There are rules for the element output order.
- Spaces, tabs, and line feeds serve as the delimiters when enumerating two or more numerical values in an element.

3.3 Attributes

3.3.1 Naming Rules for Attributes

- Attribute names are assembled from English words made from lower-case letters of the alphabet and the underscore character (_). The names do not include uppercase letters or Japanese romaji expressions.
- Some element names are abbreviated. The following list is presented alphabetically:

color	clr
frames per second	fps
index	idx
interpolation	interp
matrix	mtx
maximum	max
minimum	min
normal	nrm
position	pos
previous	prev
primitive	prim
red green blue	rgb
texture	tex
vertex	vtx
width height distance	whd

- Some attribute names are duplicated among Intermediate files and among the elements in the same file.

3.3.2 Other Rules

- There is no set order for output of attributes.
- Attributes that are optional or conditional are noted as such in the explanations below. Attributes without special mention are always output once.

3.4 Explanation of Common Header Elements and Attributes

3.4.1 The Root Element

Description: For all Intermediate files, all information is stored inside a root element that has the same name as the file extension.

Output condition: One of these is always output.

Output example: (In this case, for imd files)

```
?xml version="1.0" encoding="Shift_JIS"?>
<imd version="1.2.0">
<head>
    ... (Omitted) ...
</head>
<body>
    ... (Omitted) ...
</body>
</imd>
```

Attributes:

Table 3-1 - The Attributes of the Root Element

Attribute Name	Explanation
version	The version of the Intermediate. Real number (x1)

Contents: The root element stores one `<head>` element, which holds information about the creation of the document, followed by one `<body>` element, which holds the document itself.

3.4.2 The `<head>` Element

Description: This element stores information about who created the document and when, and comments and other information relating to the overall document. Everything below the `<head>` element is described in a format shared by all Intermediate files.

Output condition: One of these is always output

Output example:

```
<head>
    <create user="nintendo" host="nintendo-pc" date="2003-10-31T14:25:43"
    source="n64_mario.mb"/>
    <title>Model Data for NITRO</title>
    <comment>free space </comment>
    <generator name="Maya 5.0.1 NNS_Export" version="1.2.0"/>
</head>
```

Attributes: None

Contents: Stores one of each of the following elements, in this order: `<create>`, `<title>`, `<comment>`, and `<generator>`. The `<comment>` element is optional, however.

3.4.3 The `<create>` Element

Description: Stores information regarding the creation of the document.

Output condition: One of these is always output.

Output example:

```
<head>
  <create user="nintendo" host="nintendo-pc" date="2003-10-31T14:25:43"
    source="n64_mario.mb"/>
  <title>Model Data for NITRO</title>
  <comment>free space</comment>
  <generator name="Maya 5.0.1 NNS_Export" version="1.2.0"/>
</head>
```

Attributes:**Table 3-2 - The Attributes of the <create> Element**

Attribute Name	Explanation
user	Name of user who created document. Character string (x1)
host	Name of machine used to create document. Character string (x1)
date	Date of creation. Character string (x1) . The date is written in the form: YYYY-MM-DDThh:mm:ss
source	Scene name on 3D computer graphics (3D CG) tool. The output source exists only when using a 3D CG tool.
	Character string (x1) If the scene is not saved, character string is "untitled."

Contents: None.**3.4.4 The <title> Element****Description:** A string of characters describing the contents of the document is stored in this element.**Output condition:** One of these is always output.**Output example:**

```
<head>
  <create user="nintendo" host="nintendo_pc" date="2003-10-31T14:25:43"
    source="n64_mario.mb"/>
  <title>Model Data for NITRO</title>
  <comment>free space</comment>
  <generator name="Maya 5.0.1 NNS_Export" version="1.2.0"/>
</head>
```

Attributes: None**Contents:** Character string**3.4.5 The <comment> Element****Description:** Comments about the given document are stored here.**Output condition:** This element is optional.

Output example:

```
<head>
  <create user="nintendo" host="nintendo_pc" date="2003-10-31T14:25:43"
    source="n64_mario.mb"/>
  <title>Model Data for NITRO</title>
  <comment>Free space. Japanese can be written here</comment>
  <generator name="Maya 5.0.1 NNS_Export" version="1.2.0"/>
</head>
```

Attributes: None**Contents:** Arbitrary character string**Note:** Single-byte alphanumeric characters as well as double-byte characters and single-byte kana characters can be stored.

3.4.6 The <generator> Element

Description: Stores information regarding the document generator.**Output condition:** One of these is always output.**Output example:**

```
<head>
  <create user="nintendo" host="nintendo_pc" date="2003-10-31T14:25:43"
    source="n64_mario.mb"/>
  <title>Model Data for NITRO</title>
  <comment>free space</comment>
  generator name="Maya 5.0.1 NNS_Export" version="1.2.0"/>
</head>
```

Attributes:**Table 3-3 - The Attributes of the <generator> Element**

Attribute Name	Explanation
name	Name of program that output the Intermediate file. Character string (x1)
version	Version of program that output the Intermediate file. Character string (x1)

Contents: None.

3.4.7 The <body> Element

Description: Stores the information for the document itself.**Output condition:** One of these is always output

Output example:

```
<?xml version="1.0" encoding="Shift_JIS"?>
<imd version="1.0">
<head>
    ...Omitted...
</head>
<body>
    ...Omitted...
</body>
</imd>
```

Attributes: None

Contents: The `<body>` element stores a number of elements in a structure that differs depending on the Intermediate file type.

3.5 Other Rules

- Rotate Values

All Rotate values are output in degrees.

- Floating-Point Value

Note: The floating-point value output by the Intermediate File is **converted from a fixed-point value in accordance with NITRO specifications**.

When using NITRO geometry commands, change the fixed-point value according to each command specification. For example, the xyz value of the `<pos_s>` element corresponding to the NITRO geometry command `VertexShort` uses a value multiplied by 64 and rounded to the nearest integer. In a similar manner, the value of the `translate` attribute of the `<node>` element, in accordance with the `Translate` command specifications, uses a value multiplied by 4096 and rounded to the nearest integer.

4 Intermediate File for Model Data

4.1 Model Data Files (imd)

4.1.1 Overview of imd Files

The Model Data file (`imd` or Intermediate Model Data file) stores vertex information (position coordinates, normal vector, vertex color, and texture coordinates), polygon information, parent/child structures, and other information required to display 3D models, such as material, texture image, and texture palette data.

The `imd` file can be output in one of two formats: **direct** format, in which the vertex data is optimized for NITRO, and **index** format, in which vertex position coordinates and vertex color values are easier to manipulate. We recommend that you use the **direct** format except when performing special processes, such as manipulating vertex coordinates to change the form of Shape animation and vertex color animation.

Note: It is not yet decided whether Intermediate files will support Shape animation and vertex color animation.

Conforming to the NITRO specifications, the `imd` file maintains vertex position coordinates shifted so the values are -8.0 or larger but less than 8.0 . In order to properly display the shape of the model, after performing the Scale, Rotate, and Translate calculations, either shift the 3x3 section of the matrix by the `pos_scale` value output to the `imd` file in the position coordinate matrix, or multiply the position coordinate matrix by the scale matrix.

4.1.2 The Basic Element Structure of imd Files

<imd>	...Root
<head>	...Information regarding overall document
... (omitted) ...	
</head>	
<body>	...The data itself
<original create />	...Information on the environment in which the intermediate file was created
<original generator />	...Information on the tool which created the intermediate file
<model info />	...Model information
<box test />	...Box test information
<vtx_pos_data>	...Vertex position coordinate array for index format
<vtx_color_data>	...Vertex color array for index format
<tex_image_array>	...Texture image array
<tex_image>	...Texture image information
<bitmap />	...Image data
<tex4x4_palette_idx />	...Palette data for 4x4 texel compressed textures
</tex_image>	
</tex_image_array>	
<tex_palette_array>	...Texture palette array
<tex_palette />	...Texture palette information
</tex_palette_array>	
<material_array>	...Material array
<material />	...Material information
</material_array>	
<envelope>	...Weighted envelope information
<weight />	...Envelope weight data
<node_idx />	...Envelope node data
</envelope>	
<matrix_array>	...Matrix array
<matrix />	...Matrix information
</matrix_array>	
<polygon_array>	...Polygon group array
<polygon />	...Polygon group information
<mtx_prim>	...Matrix and polygon information
<mtx_list />	...Matrix list information
<primitive_array>	...Polygon array
<primitive>	...Polygon information
<mtx />	...Matrix setting
<tex />	...Texture coordinates setting
<nrm />	...Normal vector setting
<clr />	...Vertex color setting
<clr_idx />	...Vertex color setting (for index format)
<pos_s />	...Vertex coordinate setting
<pos_xy />	...Vertex coordinate setting
<pos_xz />	...Vertex coordinate setting
<pos_yz />	...Vertex coordinate setting
<pos_xyz />	...Vertex coordinate setting
<pos_diff />	...Vertex coordinate setting
<pos_idx />	...Vertex coordinate setting (for index format)
</primitive>	
</primitive_array>	
</mtx_prim>	
</polygon>	
</polygon_array>	
<node_array>	...Node array
<node>	...Node information
<display />	
</node>	
</node_array>	
<output_info />	...Output information
<ex_nns_3dme>	...Information specific to 3D Material Editor
... (omitted) ...	
</ex_nns_3dme>	
</body>	
</imd>	

4.1.3 Description of imd Elements and Attributes

4.1.3.1 The <original_create> Element

Description: Holds information concerning the environment that existed when the intermediate file was created. Resaving the file with the 3D Material Editor or a similar program will not alter this information.

Output condition: One of these is always output.

Output example:

```
<original_create user="nintendo" host="NINTENDO-PC"
                    date="2004-12-13T09:29:20" source="mario_wait"
/>
```

Attributes:

Table 4-1 - The Attributes of the <original_create> Element

Attribute Name	Description
user	The name of the user who created the file (one character string). Will be "unknown" if the user name is unclear.
host	The name of the machine that created the file (one character string). Will be "unknown" if the machine name is unclear.
date	Date created. One character string. Format: YYYY-MM-DDThh:mm:ss. Will be "unknown" if the creation date is unclear.
source This only exists if the output source is a 3DCG tool.	The name of the scene in the 3DCG tool (one character string). Will be "untitled" if no scene has been saved.

Contents: None

4.1.3.2 The <original_generator> Element

Description: Holds information on the tool that created the intermediate file. Resaving the file with the 3D Material Editor or a similar program will not alter this information.

Output condition: One of these is always output.

Output example:

```
<original_generator name="SoftimageXSI 4.0 NNS_Export"
                    version="1.5.0.20041213"
/>
```

Attributes:**Table 4-2 - The Attributes of the <original_generator> Element**

Attribute Name	Description
name	Name of the program that exported the intermediate file (one character string). Will be "unknown" if the name of the program is unclear.
version	Version of the program that exported the intermediate file (one character string). Will be "unknown" if the version of the export program is unclear.

Contents: None**4.1.3.3 The <model_info> Element****Description:** Stores information regarding overall model data.**Output condition:** One of these is always output.**Output example:**

```
<model_info
  pos_scale="0"
  scaling_rule="maya"
  vertex_style="direct"
  magnify="1.000000"
  tool_start_frame="1"
  tex_matrix_mode="maya"
  compress_node="none" node_size="31 31"
  compress_material="on" material_size="34 7"
  output_texture="used"
  force_full_weight="off"
  use_primitive_strip="on"
/>
```

Attributes:**Table 4-3 - The Attributes of the <model_info> Element**

Attribute Name	Description
pos_scale	<p>The shift value used to multiply the vertex position coordinates in NITRO. Integer 0 or larger (x1)</p> <p>When the CPU performs matrix computations, the 3x3 elements of the model matrix are shifted by this value.</p> <p>When the Geometry Engine performs matrix computations, the model matrix is multiplied by the $(1 \ll \text{pos_scale})$ multiple of the scale matrix. Also, refer to the description of the imd overview.</p>
scaling_rule	<p>The computation method for scaling. Character string [standard / maya / si3d]</p> <p>Computation methods are shown below for nodes with the following structures: parent (a), child (b), and grandchild (c).</p> <p>standard: Regular matrix computation $(Sc * Rc * Tc * Sb * Rb * Tb * Sa * Ra * Ta)$</p> <p>maya: Computation with Maya's Segment Scale Compensate taken into consideration</p> <p>si3d: SOFTIMAGE's matrix computation method</p> <p>For details about these methods, see Node Matrix Computation Methods</p>
vertex_style	<p>The method for outputting vertex position coordinates / vertex color. Character string [direct / index]</p> <p>direct: Elements corresponding to NITRO commands stored in <polygon></p> <p>index: Index numbers stored in <polygon>, and the actual data stored in other elements.</p> <p>Note: The index format is an expanded format for future use when Intermediate files support vertex color animation and shape animation. You should normally specify the direct format, which outputs data in a format optimized for NITRO.</p>
magnify	<p>The scale factor to use on the overall model when imd is output from 3D CG tool. Real number value (x1)</p> <p>Information used to confirm settings when outputting intermediate files.</p> <p>It is unnecessary to use this value on NITRO because it is reflected in the vertex position coordinates, Translate values, and other data in the intermediate file</p>
tool_start_frame	<p>The StartFrame specified when file is output from the 3D CG tool. Integer (x1)</p> <p>This is information for confirmation of settings at the time of the Intermediate File output.</p> <p>The character pose (matrix elements) stored in imd is either the pose during StartFrame or the pose during the envelope configuration. Also, the conditions during StartFrame are stored along with the visibility, etc., stored in each of the imd nodes.</p>

Attribute Name	Description
tex_matrix_mode	<p>The computation method for texture matrices. Character string [either: maya / si3d / xsi / 3dsmax] maya: Maya computation method si3d: SOFTIMAGE 3D computation method xsi: SOFTIMAGE XSI computation method 3dsmax: 3ds max computation method For details about these different methods, see Texture Matrix Computation Methods.</p>
compress_node	<p>The compression state of the node for output. Character string [none / cull / merge / unite / unite_combine] none: Output with same node structure as on the tool cull: Output after deleting the nodes not needed for displaying the model merge: In addition to the cull process, merge all node matrices that can be merged. unite: Unite all nodes into a single node. (This is mainly used for topographic data) unite_combine: Unite all nodes into a single node, and also unite all <code><polygon></code> elements that display the same <code><material></code> element.</p>
node_size	<p>The number of nodes when uncompressed and after compression. Integer 1 or larger (x2) When <code>compress_node</code> = none, the number of uncompressed nodes is listed twice.</p>
compress_material	<p>The material's compression state. Character string [off / on] off: Output the same number of materials as on the tool. on: Unite all materials with the identical settings into a single material.</p>
material_size	<p>The number of materials when uncompressed and compressed. Integer 1 or larger (x2) When <code>compress_material</code> = off, the number of uncompressed materials is listed twice.</p>
output_texture	<p>The conditions for outputting texture data from 3D CG tool to imd. Character string [used / all] used: Only output textures that are used by models output to imd. all: Output all textures in the 3D CG tool scene to imd.</p>
force_full_weight	<p>Flag that determines whether or not the full weighted envelope is forcibly used when the vertex that set the weighted envelope is detected. (The node with the most weighted is set to 100% in 3D CG tools). Character string [off / on] off: If weighted envelope data exists, output that information on: Forcibly treat weighted envelope data as full weight and output</p>
use_primitive_strip	<p>Flag that determines whether or not polygon strips are being used. Character string [either off or on] off: Polygon strips are not used, and output is in either triangular or quadrilateral polygons. on: Output using triangular and quadrilateral polygon strips as much as possible, then output those polygons that could not be included in the strips. Though this does not appear to differ from the off setting, it does reduce the number of processing vertices.</p>

Contents: None

4.1.3.4 The <box_test> Element

Description: Stores information for testing whether the model sits inside the view volume.

Corresponds to the NITRO command BoxTest.

Output condition: Output only when there are polygons.

Output example:

```
<box_test pos_scale="2" xyz="-1.197510 -0.009277 1.412354"
  whd="2.458740 4.608398 2.441406"/>
```

Attributes:

Table 4-4 - The Attributes of the <box_test> Element

Attribute Name	Description
pos_scale	Special shift value for BoxTest. Integer 0 or larger (x1) The 3x3 elements of the current position coordinate matrix are shifted by this value before the NITRO command BoxTest is sent.
xyz	The lower-right front x, y, and z coordinates of the box in which the entire model fits. Real number (x3) Corresponds to the X, Y, and Z coordinates of the NITRO command BoxTest
whd	The width, depth, and height of the box in which the entire model fits. Real number (x3) Corresponds to the width, depth, and height of the NITRO command BoxTest

Contents: None

4.1.3.5 The <vtx_pos_data> Element

Description: Stores the vertex coordinate data.

Output condition: Output only when there are polygons and the <model_info> element's vertex_style attribute is *index*.

Output example:

```
<vtx_pos_data pos_size="346">
  2.406982 -0.030518 0.998779
  2.897705 0.147705 0.010742
  ... (Omitted) ...
  1.701172 -0.427734 -2.048096
</vtx_pos_data>
```

Attributes:

Table 4-5 - The Attributes of the <vtx_pos_data> Element

Attribute Name	Description
pos_size	The number of 3D coordinates stored in the element. Integer 3 or larger (x1) Note: The actual number of data sets stored in the element is (pos_size x3) sets.

Contents: Stores (pos_size x3) real numbers, each of which has a value of -8.0 or larger but less than 8.0.

4.1.3.6 The <vtx_color_data> Element

Description: Stores vertex color data

Output condition: Output only when there are polygons, the <model_info> element's vertex_style attribute is *index*, and vertex color has been configured.

Output example:

```
<vtx_color_data color_size="4">
  0 0 31
  0 31 1
  31 31 0
  31 0 0
</vtx_color_data>
```

Attributes:

Table 4-6 - The Attributes of the <vtx_color_data> Element

Attribute Name	Description
color_size	The number of sets of RGB data stored in the element. <i>Integer 1 or larger (x1)</i> Note: The actual number of data sets stored in the element is (color_size x3) sets.

Contents: Stores (color_size x3) integers, each with a value between 0 and 31.

In accordance with the NITRO specifications, each RGB shade 0.0-1.0 is normalized to an integer between 0 and 31.

4.1.3.7 The <tex_image_array> Element

Description: Stores the image data for all textures

Output condition: Only output when there are textures

Output example:

```
<tex_image_array size="2">
  <tex_image index="0" ... (Omitted) ...>
    ... (Omitted) ...
  </tex_image>
  <tex_image index="1" (Omitted) ...>
    ... (Omitted) ...
  </tex_image>
</tex_image_array>
```

Attributes:

Table 4-7 - The Attributes of the <tex_image_array > Element

Attribute Name	Description
size	The number of <tex_image> elements stored as contents of this element. <i>Integer 1 or larger (x1)</i>

Contents: Stores size number of <tex_image> elements.

The <tex_image>elements are consecutively stored in alphanumerical order (a→z and 0→9).

4.1.3.8 The <tex_image> Element

Description: Stores the texture's image data.

Output condition: Only output when there are textures.

Output example:

```

<tex_image_array size="1">
  <tex_image index="0" name="p.1"
    width="64" height="64"
original_width="64" original_height="64"
    format="palette256" color0_mode="color"
    palette_name="p.1_pl"
    path="D:/data/TextureImage/number_pic/p.1.pic"
  >
    <bitmap size="2048">
      ... (Omitted) ...
    </bitmap>
  </tex_image>
</tex_image_array>

```

Attributes:

Table 4-8 - The Attributes of the <tex_image> Element

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
name	The name of the image (texture picture file name). Character string (x1)
width	<p>The image width.</p> <p>Integer [8 / 16 / 32 / 64 / 128 / 256 / 512 / 1024]</p> <p>Corresponds to "texture size" in the NITRO command TexImageParam</p> <p>If the original image width is not a size used by NITRO (8, 16, 32, 64, 128, 256, 512, or 1024), the right side's texel color is padded to make the image a proper NITRO width.</p>
height	<p>The image height.</p> <p>Integer [8 / 16 / 32 / 64 / 128 / 256 / 512 / 1024]</p> <p>Corresponds to "texture size" in the NITRO command TexImageParam</p> <p>If the original image height is not a size used by NITRO (8, 16, 32, 64, 128, 256, 512, or 1024), the bottom side's texel color is padded to make the image a proper NITRO height.</p>
original_width	The width of the texture picture file. Integer 1 or larger (x1)
original_height	The height of the texture picture file. Integer 1 or larger (x1)
format	<p>The format of the texture.</p> <p>Character string [palette4 / palette 16 / palette 256 / tex4x4 / a3i5 / a5i3 / direct]</p> <p>palette4: 4-color palette texture</p> <p>palette16: 16-color palette texture</p> <p>palette256: 256-color palette texture</p> <p>tex4x4: 4x4 texel compressed texture</p> <p>a3i5: A3I5 translucent texture</p> <p>a5i3: A5I3 translucent texture</p> <p>direct: Direct texture</p> <p>Corresponds to "texture format" in the NITRO command TexImageParam</p>

Attribute Name	Description
color0_mode Exists only when the format is palette4, palette16, or palette 256.	Flag indicating whether to use palette color 0 as is or as a transparent color. Character string [color / transparency] color: Use the color information as is transparency: Use as the transparent color (alpha = 0) Corresponds to the Color 0 setting value enable flag for the NTIRO command TexImageParam
palette_name Exists only when the format is not direct.	The name of the palette set with the texture image file (corresponds to the name of the <tex_palette> element). Character string (x1)
path This is optional	The PC path for the original texture picture file. Character string (x1) Note: Delimit folders using the slash (/) and not the backslash (\) sign.

Contents: Stores one <bitmap> element.

When the format is tex4x4, the <tex4x4_palette_idx> element is stored after the <bitmap> element.

4.1.3.9 The <bitmap> Element

Description: Stores the actual data of the image

Output condition: Only output when there are textures

Output example:

```

<tex_image_array size="1">
  <tex_image index="0" name="p.1"
    width="64" height="64"
original_width="64" original_height="64"
    format="palette256" color0_mode="color"
    palette_name="p.1_pl"
    path="D:/texture/p.1.pic"
  >
    <bitmap size="2048">
      1f1f 1f1f 1f1f 1f1f 1f1f 1f1f 1f1f 1f1f
        ... (Omitted) ...
      1f1f 1f1f 1f1f 1f1f 1f1f 1f1f 1f1f 1f1f
    </bitmap>
  </tex_image>
</tex_image_array>

```

Attributes:

Table 4-9 - The Attributes of the <bitmap> Element

Attribute Name	Description
size	The amount of data stored in the element. Integer (x1)

Contents: When the <tex_image> element's attribute is tex4x4, stores size amount of data in 32-bit unsigned hexadecimal. When the attribute is something other than tex4x4, stores size amount of data in 16-bit unsigned hexadecimal.

Note: Hexadecimal notation is not case-sensitive.

4.1.3.10 The <tex4x4_palette_idx> Element

Description: Stores the actual data of the palette index for 4x4 texel compressed texture

Output condition: Only output when there are 4x4 texel compressed textures

Output example:

```
<tex_image index="6" name="tex4x4_cmp2"
    idth="64" height="64"
original_width="64" original_height="64"
    format="tex4x4" color0_mode="color"
palette_name="tex4x4_cmp2_pl"
    path="D:/texture/tex4x4_cmp2.tga"
>
  <bitmap size="256">
    55aaa855 95ba002d 545e5e5c 2d555555
    ... (Omitted) ...
    0ff05575 0affd5f5 002bff57 2bbf5555
  </bitmap>
  tex4x4_palette_idx size="256">
    c000 c001 c002 c003 c004 c005 c006 c007
    ... (Omitted) ...
    c010 c011 c012 c013 c014 c015 c016 c017
  </tex4x4_palette_idx>
</tex_image>
```

Attributes:

Table 4-10 - The Attributes of the <tex4x4_palette_idx> Element

Attribute Name	Description
size	The amount of data stored in the element. Integer (x1)

Contents: Stores *size* amount of data in 16-bit unsigned hexadecimal.

Note: Hexadecimal notation is not case-sensitive.

4.1.3.11 The <tex_palette_array> Element

Description: Stores all of the texture palette data

Output condition: Only output when there are textures that employ palette textures.

Output example:

```
<tex_palette_array size="2">
  <tex_palette index="0" name="p.1_pl" color_size="16">
    001f 043f 085f 0c7f 109f 14bf 18df 1cff
    631f 673f 6b5f 6f7f 739f 77bf 7bdf 7fff
  </tex_palette>
  <tex_palette index="1" name="p.2_pl" color_size="32">
    ... (Omitted) ...
  </tex_palette>
</tex_palette_array>
```

Attributes:**Table 4-11 - The Attributes of the <tex_palette_array> Element**

Attribute Name	Description
size	The number of <tex_palette> elements stored in this element. <i>Integer 1 or larger (x1)</i>

Contents: Stores *size* number of <tex_palette> elements.

<tex_palette>elements are stored consecutively by palette name in alphanumerical order (a→z and 0→9).

4.1.3.12 The <tex_palette> Element

Description: Stores the actual data of the palette

Output condition: Only output when there are textures that employ palette textures.

Output example:

```
<tex_palette_array size="1">
  <tex_palette index="0" name="p.1_pl" color_size="16">
    1f 043f 085f 0c7f 109f 14bf 18df 1cff
    631f 673f 6b5f 6f7f 739f 77bf 7bdf 7fff
  /tex_palette>
</tex_palette_array>
```

Attributes:**Table 4-12 - The Attributes of the <tex_palette> Element**

Attribute Name	Description
index	Index number. <i>Integer 0 or larger (x1)</i>
name	Palette name. <i>Character string (x1)</i>
color_size	Number of colors in palette. <i>Integer 4 or multiple of 8</i>

Contents: Stores *color_size* amount in 16-bit unsigned hexadecimal.

Note: Hexadecimal notation is not case-sensitive.

4.1.3.13 The <material_array> Element

Description: Stores all of the material data.

Output condition: Only output when materials exist

Output example:

```
<material_array size="2">
  <material index="0" ... (Omitted) ... />
  <material index="1" ... (Omitted) ... />
</material_array>
```

Attributes:**Table 4-13 - The Attributes of the <material_array> Element**

Attribute Name	Description
size	The number of <material> elements stored in this element. <i>Integer 1 or larger (x1)</i>

Contents: Stores *size* number of <material> elements.

The <material> elements are consecutively stored by material name in alphanumerical order (a→z, 0→9).

4.1.3.14 The <material> Element

Description: Stores material data

Output condition: Only output when material exists

Output example:

```

<material_array size="1">
  <material index="0" name="lambert2"
    light0="on" light1="off" light2="off" light3="off"
    face="front"
    alpha="31"
    wire_mode="off"
    polygon_mode="modulate"
    polygon_id="0"
    fog flag="off"
    depth_test_decals="on"
    translucent_update_depth="on"
    render_1_pixel="on"
    far_clipping="on"
    diffuse="25 25 25"
    ambient="0 0 0"
    specular="0 0 0"
    emission="0 0 0"
    shininess table flag="off"
    tex_image_idx="0" tex_palette_idx="0"
    tex_tiling="repeat repeat"
    tex_scale="1.000000 1.000000"
    tex_rotate="0.000000"
    tex_translate="0.000000 0.000000"
    tex_gen_mode="nrm"
    tex_gen_st_src="material"
    tex_effect_mtx="1.000000 0.000000 0.000000 0.000000
                    0.000000 1.000000 0.000000 0.000000
                    0.000000 0.000000 1.000000 0.000000
                    0.000000 0.000000 0.000000 1.000000"
  />
</material_array>

```

Attributes:**Table 4-14 - The Attributes of the <material> Element**

Attribute Name	Description
index	Index number. <i>Integer 0 or larger(x1)</i>

Attribute Name	Description
name	Material name. Character string (x1)
light0	Light0 enable flag. Character string [off / on] Corresponds to "light enable flag 0" of the NITRO command PolygonAttr.
light1	Light1 enable flag. Character string [off / on] Corresponds to "light enable flag 1" of the NITRO command PolygonAttr.
light2	Light2 enable flag. Character string [off / on] Corresponds to "light enable flag 2" of the NITRO command PolygonAttr.
light3	Light3 enable flag. Character string [off / on] Corresponds to "light enable flag 3" of the NITRO command PolygonAttr.
face	The face on which to draw polygons. Character string [front / back / both] Front: Only display the front face Back: Only display the back face Both: Display both faces Corresponds to the "polygon drawing face" of the NITRO command PolygonAttr.
alpha	Polygon alpha. $0 \leq \text{integer} \leq 31$ (x1) Corresponds to alpha value of the NITRO command PolygonAttr. When this value is 0, skips the rendering of the polygon assigned to this material and does not display anything.
wire_mode	Wireframe display mode. Character string [off / on] off: Use alpha value. When alpha = 0, hide. on: Display with normal wireframe (alpha = 0)
polygon_mode	Polygon drawing mode Character string [modulate / decal / toon_highlight / shadow] Corresponds to "polygon mode" of the NITRO command PolygonAttr.
polygon_id	Polygon ID. $0 \leq \text{integer} \leq 63$ (x1) Corresponds to "polygon ID" of the NITRO command PolygonAttr.
fog_flag	Fog enable flag. Character string [off / on] Corresponds to the "fog enable flag" of the NITRO command PolygonAttr.
depth_test_decal	Depth test condition. Character string [off / on] off: Render if the depth of the fragment is less than the depth of the depth buffer. on: Render if the depth of the fragment is equal to the depth of the depth buffer (decal polygon). This is equivalent to the depth condition in the NITRO command PolygonAttr.

Attribute Name	Description
translucent_update_depth	Enables updating the depth value of translucent polygons. Character string [off / on] off: Do not update the depth buffer when rendering translucent polygons. on: Update the depth buffer when rendering translucent polygons. This is equivalent to the flag that enables updating the translucent polygon depth value in the NITRO command PolygonAttr.
render_1_pixel	One-pixel (dot) polygon rendering specification. Character string [off / on] off: Do not render a polygon if it is only one pixel (dot). on: Render a polygon, even if it is only one pixel (dot). This is equivalent to specifying one-dot polygon display in the NITRO command PolygonAttr.
far_clipping	Specifies display for polygons that intersect the far plane. Character string [off / on] off: Delete if it intersects the far plane. on: Clip if it intersects the far plane. This is equivalent to specifying the display of polygons that intersect the far plane in the NITRO command PolygonAttr.
diffuse	Diffuse reflection color (R,G,B). 0 ≤ integer ≤ 31 (x3) Corresponds to "diffuse" of the NITRO command MaterialColor0.
ambient	Ambient reflection color (R,G,B). 0 ≤ integer ≤ 31 (x3) Corresponds to "ambient" of the NITRO command MaterialColor0.
specular	Specular reflection color (R,G,B). 0 ≤ integer ≤ 31 (x3) Corresponds to "specular" of the NITRO command MaterialColor1.
emission	Emission color (R,G,B). 0 ≤ integer ≤ 31 (x3) Corresponds to "emission" of the NITRO command MaterialColor1.
shininess_table_flag	Specular Reflection Brightness Flag. Character string [off / on] Corresponds to the "shininess table enable flag" of the NITRO command MaterialColor1.
tex_image_idx	The texture's image number. Integer -1 or larger (x1) When a texture is not mapped, value takes -1.
tex_palette_idx	The texture's palette number. Integer -1 or larger (x1) When a texture that uses a palette is not mapped, value takes -1.
tex_tiling Exists when tex_image_idx is 0 or larger	The tiling method for the texture. Character string [clamp / repeat / flip] (x2) Corresponds to "flip - repeat" of the NITRO command TexImageParam. clamp: Do not repeat; Do not flip repeat: Do not flip flip: Flip

Attribute Name	Description
<code>tex_scale</code> Exists when <code>tex_image_idx</code> is 0 or larger	The ScaleS and ScaleT values configured in the texture matrix Real number (x2)
<code>tex_rotate</code> Exists when <code>tex_image_idx</code> is 0 or larger	The rotate value configured in the texture matrix $-180.0 \leq \text{real number} < 180.0$ (x1)
<code>tex_translate</code> Exists when <code>tex_image_idx</code> is 0 or larger	The TranslateS and TranslateT values configured in the texture matrix Real number (x2)
<code>tex_gen_mode</code> Exists when <code>tex_image_idx</code> is 0 or larger	<p>Texture coordinate transformation mode. Character string [none / tex / nrm / pos]</p> <p>Corresponds to "texture coordinate transformation mode" of NITRO command <code>TexImageParam</code>.</p> <p>none: No texture coordinate conversion tex: The TexCoord source nrm: The Normal source pos: The Vertex source</p> <p>Designate nrm when performing environmental mapping, and pos when performing projection mapping.</p> <p>Note: The <code><nrm></code> element inside the <code><polygon></code> element is usually stored each time the direction of the normal vector changes, but for <code>tex_gen_mode="nrm"</code> and <code>tex_gen_st_src="polygon"</code>, it can also be stored after a change in the texture coordinates. (A <code><nrm></code> element is always stored between a <code><tex></code> element and a <code><pos_***></code> element.)</p>
<code>tex_gen_st_src</code> Exists when <code>tex_gen_mode</code> is <i>nrm</i> or <i>pos</i>	<p>A flag that determines whether to output the texture coordinates (the <code><tex></code> element) inside the corresponding <code><polygon></code> element when either <code>tex_gen_mode="nrm"</code> (the texture coordinate conversion mode is a Normal source) or when <code>tex_gen_mode="pos"</code> (the texture coordinate conversion mode is a Vertex source).</p> <p>The flag is either "polygon" or "material":</p> <p>polygon: The <code><tex></code> element is output inside the corresponding <code><polygon></code> element. Special mapping expressions are possible, such as transformation of standard attached textures via normal vectors or vertex coordinates.</p> <p>material: The <code><tex></code> element is not output inside the corresponding <code><polygon></code> element. Use this option when performing general environmental and projection mapping.</p>

Attribute Name	Description
tex_effect_mtx Exists when tex_gen_mode is <i>nrm</i> or <i>pos</i> .	<p>The elements of a 4x4 matrix that have influence when displaying a texture with environmental or projection mapping. Real Number (x16)</p> <p>The values correspond to the arrangement in the following 4x4 matrix:</p> <pre> 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 </pre> <p>Note: Refer to "Using tex_effect_mtx" in the "Supplemental Information about imd"</p>

Contents: None

4.1.3.15 The <envelope> Element

Description: Stores data for weighted envelope

For information about the matrix reference procedure, see "Vertex Position Coordinates Transformation Matrix".

Output condition: Only output when weighted envelope has been configured

Output example:

```

<envelope>
  <weight size="11">
    60 40 50 42 8 50 50 60 40 50
    50 50
  </weight>
  <node_idx size="11">
    8 9 8 9 6 8 9 5 6 5
    6 9
  </node_idx>
</envelope>

```

Attributes: None

Contents: Stores one <weight> element and one <node_idx> element in this order

4.1.3.16 The <weight> Element

Description: Stores the weighted envelope's weight information

Output condition: Only output when weighted envelope has been configured

Output example:

```
<envelope>
  <weight size="11">
    60 40 50 42 8 50 50 60 40 50
    50
  </weight>
  <node_idx size="11">
    8 9 8 9 6 8 9 5 6 5
    6
  </node_idx>
</envelope>
```

Attributes:

Table 4-15 - The Attributes of the <weight> Element

Attribute Name	Description
size	The amount of data stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of integers, where integers take value between 1 and 99 (Unit = %. Integer values must add up to 100)

Note: The values output here correspond to those of <node_idx>

4.1.3.17 The <node_idx> Element

Description: Stores the weighted envelope's node information

Output condition: Only output when the weighted envelope has been configured

Output example:

```
<envelope>
  <weight size="11">
    60 40 50 42 8 50 50 60 40 50
    50
  </weight>
  node_idx size="11">
    8 9 8 9 6 8 9 5 6 5
    6
  </node_idx>
</envelope>
```

Attributes:

Table 4-16 - The Attributes of the <node_idx> Element

Attribute Name	Description
size	The number of nodes stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of integers, where each integer takes a value of 0 or larger (the node number)

Note: The values output here match with those of <weight>.

4.1.3.18 The <matrix_array> Element

Description: Stores all matrix information necessary for drawing

Output condition: Only output when polygons exist

Output example:

```
<matrix_array size="3">
  <matrix index="0" mtx_weight="2" envelope_head="0"/>
  <matrix index="1" mtx_weight="1" node_idx="8"/>
  <matrix index="2" mtx_weight="1" node_idx="9"/>
</matrix_array>
```

Attributes:

Table 4-17 - The Attributes of the <matrix_array> Element

Attribute Name	Description
size	The number of <matrix> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <matrix> elements

Note: Indexes and stores the *mtx_weight* values in descending order, and the *envelope_head* and *node_idx* values in ascending order.

4.1.3.19 The <matrix> Element

Description: Stores the matrix information used for drawing

For information about the matrix reference procedure, see "Vertex Position Coordinates Transformation Matrix."

Output condition: Only output when polygons exist

Output example:

```
<matrix_array size="4">
  matrix index="0" mtx_weight="3" envelope_head="2"/>
  <matrix index="1" mtx_weight="2" envelope_head="0"/>
  <matrix index="2" mtx_weight="1" node_idx="9"/>
  <matrix index="3" mtx_weight="1" node_idx="10"/>
</matrix_array>
```

Attributes:

Table 4-18 - The Attributes of the <matrix> Element

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
mtx_weight	The number of weighted nodes. Integer 1 or larger (x1) When 1: Depends on the specific node matrix When 2 or more: Depends on multiple node matrices (weighted envelope)
node_idx When mtx_weight is 1	This value is used as the vertex transformation matrix as calculated by the <node> element that becomes the index. Integer 0 or larger (x1)

Attribute Name	Description
envelope_head When <code>mtx_weight</code> is 2 or more	Starting position of data in <envelope>. Integer 0 or larger (x1) References <weight>, <node_idx> elements. Envelope calculations are performed with node number and weight value at the n th position from <code>envelope_head</code> , where n = the number of weighted nodes (<code>mtx_weight</code>).

Contents: None

4.1.3.20 The <polygon_array> Element

Description: Stores all polygon information.

Output condition: Only output when polygons exist

Output example:

```
<polygon_array size="2">  
  <polygon index="0" name="polygon0" ... (Omitted) ...  
    ... (Omitted) ...  
</polygon>  
  <polygon index="1" name="polygon1" ... (Omitted) ...  
    ... (Omitted) ...  
</polygon>  
</polygon_array>
```

Attributes:

Table 4-19 - The Attributes of the <polygon_array> Element

Attribute Name	Description
size	The number of <polygon> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores `size` number of <polygon> elements.

4.1.3.21 The <polygon> Element

Description: Stores polygon information.

Output condition: Only output when polygons exist

Output example:

```
polygon index="0" name="polygon0"  
  vertex_size="40" polygon_size="1" triangle_size="20" quad_size="1"  
  volume_min="-5.000000 -5.000000 5.000000"  
  volume_max="5.000000 5.000000 5.000000"  
  volume_r="8.660254"  
  mtx_prim_size="1"  
  nrm_flag="off" clr_flag="off" tex_flag="on">  
  <mtx_prim index="0">  
    <mtx_list size="1">0</mtx_list>  
    <primitive_array size="5">  
      ... (Omitted) ...  
    </primitive_array>  
  </mtx_prim>  
</polygon>
```

Attributes:**Table 4-20 - The Attributes of the <polygon> Element**

Attribute Name	Description
index	Index number. Integer 0 (zero) or larger (x1)
name	<p>The name of this polygon group. Character string (x1)</p> <p>Note: If created in Maya, SOFTIMAGE 3D, SOFTIMAGE XSI, takes the form “polygon + serial number”.</p> <p>Note: If created in 3ds max, become s polygon name that is on 3ds max. However, if multiple polygon names are defined with the same name, “serial number” is appended at the end of each.</p>
vertex_size	The number of vertices. Integer 3 or larger (x1)
polygon_size	The total number of polygons (number of triangular polygons + number of quadrilateral polygons). Integer 1 or larger (x1)
triangle_size	Number of triangular polygons. Integer 0 or larger (x1)
quad_size	Number of quadrilateral polygons. Integer 0 or larger (x1)
volume_min	Smallest local coordinate value in this polygon group (X, Y, Z). Real number (x3) Used for collision detection.
volume_max	Largest local coordinate value in this polygon group (X, Y, Z). Real number (x3) Used for collision detection.
volume_r	Radius of sphere that circumscribes this polygon group. Real number (x1) Used for collision detection.
mtx_prim_size	Number of <mtx_prim> elements stored in this polygon group. Integer 1 or larger (x1)
nrm_flag	Flag indicating whether <nrm> element is stored here. Character string [off / on]
clr_flag	Flag indicating whether <clr> element is stored here. Character string [off / on]
tex_flag	Flag indicating whether <tex> element is stored here. Character string [off / on]

Contents: Stores *mtx_prim_size* number of <mtx_prim> elements.

4.1.3.22 The <mtx_prim> Element

Description: Stores matrix lists and polygon data

Output condition: Only output when polygons exist

Output example:

```
<mtx_prim index="0">
<mtx_list size="2">0 1</mtx_list>
<primitive_array size="1">
... (Omitted) ...
</primitive_array>
</mtx_prim>
```

Attributes:**Table 4-21 - The Attributes of the <mtx_prim> Element**

Attribute Name	Description
index	Index number. (Integer 0 or larger) x1

Contents: One <mtx_list> element and one <primitive_array> element are stored in that order.

4.1.3.23 The <mtx_list> Element

Description: Stores the matrix information required for drawing. For information about the matrix reference procedure, see "Vertex Position Coordinates Transformation Matrix."

Output condition: Only output when polygons exist

Output example:

```
<mtx_prim index="0">
<mtx_list size="2">0 1</mtx_list>
<primitive_array size="1">
... (Omitted) ...
</primitive_array>
</mtx_prim>
```

Attributes:**Table 4-22 - The Attributes of the <mtx_list> Element**

Attribute Name	Description
size	The amount of data included in this element. $1 \leq \text{Integer} \leq 31$ (x1)

Contents: Stores the size amount of matrix numbers (index numbers to <matrix> elements) required for drawing the polygons in all the successive <primitive_array> elements.

When the envelope is not used, the size attribute must be 1, and one (integer > 0) is stored in the contents.

When the envelope is used, (size) number of (integer ≥ -1) is stored in the contents.

The 0 or greater value stored in the contents is an index to the <matrix> element. However, only when the `mtx_prim_size` attribute of the <polygon> element ≥ 2 are there times when -1 is stored in the contents of a <mtx_list> element greater than 2. Because -1 can use the matrix specified in the same location as the previous <mtx_list> as is, this means there is no need to update the stack.

4.1.3.24 The <primitive_array> Element

Description: Stores polygon information

Output condition: Only output when polygons exist

Output example:

```

<mtx_prim index="0">
<mtx_list size="2">0 1</mtx_list>
<primitive_array size="5">
<primitive index="0" type="quad_strip" vertex_size="6">
    ... (Omitted) ...
</primitive>
<primitive index="1" type="triangle_strip" vertex_size="8">
    ... (Omitted) ...
</primitive>
<primitive index="2" type="triangle_strip" vertex_size="7">
    ... (Omitted) ...
</primitive>
<primitive index="3" type="quads" vertex_size="4">
    ... (Omitted) ...
</primitive>
<primitive index="4" type="triangles" vertex_size="3">
    ... (Omitted) ...
</primitive>
</primitive_array>
</mtx_prim>

```

Attributes:

Table 4-23 - The Attributes of the <primitive_array> Element

Attribute Name	Description
size	The number of <prim> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <primitive> elements

Note: Stores *quad_strip*, *triangle_strip*, *quads*, and *triangles*, in that order, after indexing the vertices in order from the largest.

4.1.3.25 The <primitive> Element

Description: Stores polygon data

Corresponds to NITRO commands Begin / End

Output condition: Only output when polygons exist

Output example:

```

<primitive_array size="2">
  primitive index="0" type="triangle strip" vertex_size="9">
    ... (Omitted) ...
  /primitive>
<primitive index="1" type="triangle strip" vertex_size="6">
  <pos_xy xy="2.406982 -0.030518"/>
    <nrm xyz="0.492188 0.726563 0.478516"/>
    <pos_xyz xyz="1.970703 1.489746 0.651611"/>
    <nrm xyz="-0.521484 0.636719 0.568359"/>
    <pos_xyz xyz="0.433350 1.601318 0.760986"/>
    <nrm xyz="-0.544922 0.605469 -0.582031"/>
    <pos_xyz xyz="0.415771 1.597168 -0.793457"/>
    <nrm xyz="-0.949219 0.042969 -0.310547"/>
    <pos_xyz xyz="0.037354 0.813232 -0.375732"/>
    <nrm xyz="-0.441406 -0.326172 -0.835938"/>
    <pos_xyz xyz="0.677246 -0.232422 -1.155518"/>
  /primitive>
</primitive_array>

```

Attributes:**Table 4-24 - The Attributes of the <primitive> Element**

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
type	<p>The method for drawing polygons.</p> <p>Character string [triangles / quads / triangle_strip / quad_strip]</p> <p>triangles: Display as triangular polygons</p> <p>quads: Display as quadrilateral polygons</p> <p>triangle_strip: Display as a triangle polygon strip</p> <p>quad_strip: Display as a quadrilateral polygon strip</p> <p>Corresponds to the "primitive type" of the NITRO command Begin</p>
vertex_size	Number of vertices. Integer 3 or larger (x1)

Contents: The elements <mtx>, <tex>, <nrm>, <clr>, <clr_idx>, <pos_s>, <pos_xy>, <pos_yz>, <pos_xz>, <pos_xyz>, <pos_diff>, and <pos_idx> are stored multiple times in vertex list order.

Note: The <pos_*> element is output *vertex_size* number of times.

Note: **Note:** When the <model_info> element's vertex_type is *index*, the <clr_idx> element is substituted for the <clr> element and the <pos_idx> element is substituted for the <pos_*> element.

4.1.3.26 The <mtx> Element

Description: Stores the matrix list number of the referenced matrix (number of the <mtx_list> element contents)

When the CPU computes the matrix and the (full) weighted envelope model is displayed, this corresponds to the NITRO command RestoreMatrix. For information about the matrix reference procedure, see "Vertex Position Coordinates Transformation Matrix."

Output condition: Only output when polygons exist

Output example:

```

<mtx_list size="2">0 2</mtx_list>
<primitive_array size="1">
<primitive index="0" type="triangles" vertex_size="3">
    mtx idx="0"/>
    <nrm xyz="0.000000 0.998047 0.000000"/>
    <pos_xyz xyz="-7.899902 0.000000 7.899902"/>
    <pos_xy xy="7.899902 0.000000"/>
    <pos_xz xz="7.899902 -7.899902"/>
</primitive>
</primitive_array>

```

Attributes:**Table 4-25 - The Attributes of the <mtx> Element**

Attribute Name	Description
idx	The matrix number used for drawing. Integer 0 or larger (x1) References <mtx_list>

Contents: None**4.1.3.27 The <tex> Element****Description:** Stores one group of texture coordinate data (s and t)Corresponds to the NITRO command `TexCoord`**Output condition:** Only output when texture-mapped polygons are present.However, it will not be output when `tex_gen_src="material"` for the corresponding <material> element.**Output example:**

```

<primitive index="1" type="triangles" vertex_size="3">
    tex st="0.000000 0.000000"/>
    <nrm xyz="0.000000 0.000000 0.998047"/>
    <pos_s xyz="-2.500000 -2.500000 0.000000"/>
    tex st="64.000000 0.000000"/>
    <pos_xy xy="2.500000 -2.500000"/>
    tex st="0.000000 64.000000"/>
    <pos_xy xy="-2.500000 2.500000"/>
</primitive>

```

Attributes:**Table 4-26 - The Attributes of the <tex> Element**

Attribute Name	Description
st	The s and t components of the texture coordinates. Real number (x2)

Contents: None

4.1.3.28 The <nrm> Element

Description: Stores one group of normal vector data (nx, ny, nz)

Corresponds to the NITRO command Normal

Output condition: Only output when light is computed

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">
  <tex st="0.000000 0.000000"/>
  nrm xyz="0.000000 0.000000 0.998047"/>
  <pos_s xyz="-2.500000 -2.500000 0.000000"/>
  <tex st="64.000000 0.000000"/>
  <pos_xy xy="2.500000 -2.500000"/>
  <tex st="0.000000 64.000000"/>
  <pos_xy xy="-2.500000 2.500000"/>
</primitive>
```

Attributes:

Table 4-27 - The Attributes of the <nrm> Element

Attribute Name	Description
xyz	The x, y, and z components of the normal vector. (-1.0 ≤ real number ≤ 0.998047) (x3)

Contents: None

4.1.3.29 The <clr> Element

Description: Stores one group of vertex color data (r,g,b)

Corresponds to the NITRO command Color or MaterialColor0

Output condition: Output when the <model_info> element's vertex_style is *direct* and vertex color is being used

Output example:

```
<primitive index="2" type="triangles" vertex_size="3">
  clr rgb="0 1 31"/>
  <pos_xyz xyz="-7.899902 0.000000 7.899902"/>
  clr rgb="0 31 0"/>
  <pos_xy xy="7.899902 0.000000"/>
  clr rgb="31 30 0"/>
  <pos_xz xz="7.899902 -7.899902"/>
</primitive>
```

Attributes:

Table 4-28 - The Attributes of the <clr> Element

Attribute Name	Description
rgb	The r, g, and b components of vertex color. (0 < integer 0 ≤ 31) (x3)

Contents: None

4.1.3.30 The <clr_idx> Element

Description: Stores the index number to the vertex color data (<vtx_color_data> element)

Output condition: Output when the <model_info> element's vertex_style is *index* and vertex color is being used.

Output example:

```
<primitive index="2" type="triangles" vertex_size="3">
  <tex st="0.000000 0.000000"/>
  <clr_idx idx="0"/>
  <pos_idx idx="6"/>
  <tex st="16.000000 0.000000"/>
  <clr_idx idx="1"/>
  <pos_idx idx="9"/>
  <tex st="16.000000 16.000000"/>
  <clr_idx idx="2"/>
  <pos_idx idx="7"/>
</primitive>
```

Attributes:

Table 4-29 - The Attributes of the <clr_idx> Element

Attribute Name	Description
idx	The index number to the vertex color data. Integer 0 or larger (x1) The three values starting from the n th <vtx_color_data> element (where n = idx * 3) are referenced as r, g, and b.

Contents: None

4.1.3.31 The <pos_s> Element

Description: Stores 1 group of vertex position coordinates (x, y, z)

Corresponds to the NITRO command VertexShort

Output condition: Output will occur once the following conditions have been met:

- When polygons exist and vertex_style is *direct*
- The <pos_xy>, <pos_xz>, <pos_yz>, or <pos_diff> conditions do not apply
- When the lower 6 bits of 12-bit decimals are all 0, in the case that x, y, and z are each converted to fixed-point decimal values

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">
  <nrm xyz="0.000000 0.000000 0.998047"/>
  <pos_s xyz="-2.500000 -2.500000 0.000000"/>
  <pos_xy xy="2.500000 -2.500000"/>
  <pos_xy xy="2.500000 2.500000"/>
</primitive>
```

Attributes:**Table 4-30 - The Attributes of the <pos_s> Element**

Attribute Name	Description
xyz	The vertex position coordinates. $(-8.0 \leq \text{real numbers} \leq 8.0)$ (x3)

Contents: None

4.1.3.32 The <pos_xy> Element

Description: Stores the (x, y) coordinates of one group of vertex position coordinates.

Corresponds to the NITRO command VertexXY

Output condition: Output will occur once the following conditions have been met

- Output if polygons exist and `vertex_style` is *direct*.
- When the z coordinate is the same as the z coordinate value set immediately before.

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">  
  <pos_xyz xyz="2.500000 0.100000 -2.500000"/>  
  <pos_xy xy="-1.500000 -1.500000"/>  
  <pos_yz yz="3.500000 3.500000"/>  
</primitive>
```

Attributes:**Table 4-31 - The Attributes of the <pos_xy> Element**

Attribute Name	Description
xy	The x and y components of the vertex position coordinates. $(-8.0 \leq \text{real numbers} \leq 8.0)$ (x2)

Contents: None

4.1.3.33 The <pos_xz> Element

Description: Stores the (x, z) coordinates of one group of vertex position coordinates.

Corresponds to the NITRO command VertexXZ

Output condition: Output will occur once the following conditions have been met

- Output if polygons exist and `vertex_style` is *direct*.
- When the y coordinate is the same as the y coordinate value set immediately before.

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">  
  <pos_xy xy="-1.500000 -1.500000"/>  
  <pos_xz xz="2.500000 -2.500000"/>  
  <pos_yz yz="3.500000 3.500000"/>  
</primitive>
```

Attributes:**Table 4-32 - The Attributes of the <pos_xz> Element**

Attribute Name	Description
xz	The x and z components of the vertex position coordinates. (-8.0 ≤ real numbers ≤ 8.0) (x2)

Contents: None**4.1.3.34 The <pos_yz> Element****Description:** Stores the (y, z) coordinates of one group of vertex position coordinates.

Corresponds to the NITRO command VertexYZ

Output condition: Output will occur once the following conditions have been met

- Output if polygons exist and vertex_style is *direct*.
- When the x coordinate is the same as the x coordinate value set immediately before.

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">
  <pos_xy xy="-1.500000 -1.500000"/>
  <pos_xz xz="2.500000 -2.500000"/>
  pos_yz yz="3.500000 3.500000"/>
</primitive>
```

Attributes:**Table 4-33 - The Attributes of the <pos_yz> Element**

Attribute Name	Description
yz	The y and z components of the vertex position coordinates. (-8.0 ≤ real numbers ≤ 8.0) (x2)

Contents: None**4.1.3.35 The <pos_xyz> Element****Description:** Stores the (x, y, z) coordinates of one group of vertex position coordinates.

Corresponds to the NITRO command Vertex.

Output condition: Output will occur once the following conditions have been met:

- When polygons exist and vertex_style is *direct*.
- The <pos_xy>, <pos_xz>, <pos_yz>, <pos_diff>, or <pos_s> conditions do not apply.

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">
  pos_xyz xyz="1.500000 1.500000 1.500000"/>
  <pos_xz xz="2.500000 -2.500000"/>
  <pos_yz yz="3.500000 3.500000"/>
</primitive>
```

Attributes:

Table 4-34 - The Attributes of the <pos_xyz> Element

Attribute Name	Description
xyz	The x, y, and z components of the vertex position coordinates. (-8.0 ≤ real numbers ≤ 8.0) (x3)

Contents: None

4.1.3.36 The <pos_diff> Element

Description: Stores the (x, y, z) coordinates of one group of vertex position coordinates.

Corresponds to the NITRO command VertexDiff

Output condition: Output will occur once the following conditions have been met:

- When polygons exist and `vertex_style` is *direct*.
- The margin of error for the vertex position coordinates set immediately before and the x, y, and z coordinates are ? -0.125 and < 0.125

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">  
  <pos_xyz xyz="1.500000 1.500000 1.500000"/>  
  pos_diff xzz="0.000010 0.000007 0.000008"/>  
  <pos_xyz xyz="0.000000 3.500000 3.500000"/>  
</primitive>
```

Attributes:

Table 4-35 - The Attributes of the <pos_diff> Element

Attribute Name	Description
xyz	The amount of difference from the last-set vertex position coordinates. (-0.125 ≤ real numbers ≤ 0.125) (x3)

Contents: None

4.1.3.37 The <pos_idx> Element

Description: Stores the index number to the vertex position coordinate data (<vtx_pos_data> element)

Output condition: When polygons exist and `vertex_style` is *index*.

Output example:

```
<primitive index="1" type="triangles" vertex_size="3">  
  pos_idx idx="5"/>  
  <pos_idx idx="7"/>  
  <pos_idx idx="10"/>  
</primitive>
```

Attributes:**Table 4-36 - The Attributes of the <pos_idx> Element**

Attribute Name	Description
idx	Index number to the vertex position coordinate data. Integer 0 or larger (x1) The three values starting from the n^{th} <vtx_pos_data> element (where $n = \text{idx} \times 3$) are referenced as x, y, and z.

Contents: None**4.1.3.38 The <node_array> Element****Description:** Stores all node information**Output condition:** One of these is always output**Output example:**

```

<node_array size="2">
  <node index="0" name="null" ... (Omitted) ...>
    ... (Omitted) ...
  </node>
  <node index="1" name="pPlane1" ... (Omitted) ...>
    ... (Omitted) ...
  </node>
</node_array>

```

Attributes:**Table 4-37 - The Attributes of the <node_array> Element**

Attribute Name	Description
size	The number of <node> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <node> elements

Adhering to the parent/child structure on the 3D CG tool, the child node takes priority and sibling nodes are consecutively stored by node name in alphanumerical order (a→z and 0→9). The model root node must be output to the index0th <node> element.

4.1.3.39 The <node> Element**Description:** Stores node information**Output condition:** One or more of these is always output

Output example:

```

<node_array size="1">
  node index="0" name="pPlane1" kind="mesh"
    parent="-1" child="-1"
    brother_next="-1" brother_prev="-1"
    draw_mtx="on"
    scale_compensate="off"
    billboard="off"
    scale="1.000000 1.000000 1.000000"
    rotate="0.000000 0.000000 0.000000"
    translate="0.000000 0.000000 0.000000"
    visibility="on"
    display_size="1"
    vertex_size="4" polygon_size="1"
    triangle_size="0" quad_size="1"
    volume_min="-7.899902 0.000000 -7.899902"
    volume_max="7.899902 0.000000 7.899902"
    volume_r="11.172363"
  >
    <display index="0" material="0" polygon="0"/>
  /node>
</node_array>

```

Attributes:**Table 4-38 - The Attributes of the <node> Element**

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
name	Node name. Character string (x1)
Kind	Node type. Character string [null / mesh / joint / chain / effector] null: Null node (corresponds to Maya's Locator, SOFTIMAGE 3D Null) mesh: Node that has a polygon joint: Joint (bone) node chain: The main parent node of the joint effector: Node that is the end of the joint
parent	The parent node's number. Integer -1 or larger If the value is -1, the node element has no parent and is a root node.
child	The youngest child's node number (its own child and the first node when names are arranged alphabetically starting with "a"). Integer -1 or larger If the value is -1, the node has no child nodes and is itself the end node
brother_next	The next sibling's node number (the sibling nodes with the same parent node are arranged by name in alphabetical order starting with a of the next node). Integer -1 or larger If the value is -1, the node has no nextSibling.
brother_prev	The previous sibling's node number. Integer -1 or larger If the value is -1, the node has no previous sibling.
draw_mtx	Flag indicating whether the node's matrix is being used for polygon drawing. Nodes where this value is set to off become the target of node culling during node compression. Character string [off / on]

Attribute Name	Description
scale_compensate Note: <i>scaling_rule</i> is output only with Maya.	Scale computation flag set with Maya. Character string [off / on] off: Maya's segment_scale_compensate not computed on: Maya's segment_scale_compensate computed
billboard	Billboard display setting. Character string [off / on / y_on] off: Normal display on: Always face the camera y_on: Face the camera by rotating only along the global Y axis
scale	Nodal ScaleX, ScaleY, and ScaleZ values. Real numbers (x3)
rotate	Nodal RotateX, RotateY, and RotateZ values. -180.0 ≤ real numbers ≤ 180.0 (x3)
translate	Nodal TranslateX, TranslateY, and TranslateZ values. Real numbers (x3)
visibility	Show/hide display setting. Character string [off / on]
display_size	Number of <display> elements output as child elements. Integer 0 or larger (x1)
vertex_size Exists only when the node has polygons (kind=mesh).	The number of vertices. Integer 3 or larger (x1)
polygon_size Exists only when the node has polygons (kind=mesh)	The number of polygons (the number of triangular polygons + the number of quadrilateral polygons). Integer 1 or larger (x1)
triangle_size Exists only when the node has polygons (kind=mesh)	Number of triangular polygons. Integer 0 or larger (x1)
quad_size Exists only when the node has polygons (kind=mesh)	Number of quadrilateral polygons. Integer 0 or larger (x1)
volume_min Exists only when the node has polygons (kind=mesh)	The lower left inside local coordinates (x, y, z) of the circumscribed grid of all polygons assigned to the node. Real numbers (x3) Can be used for collision detection.
volume_max Exists only when the node has polygons (kind=mesh)	The upper right front local coordinates (x, y, z) of the circumscribed grid of all polygons assigned to the node. Real numbers (x3) Can be used for collision detection.
volume_r Exists only when the node has polygons (kind=mesh)	The radius of the sphere that circumscribes all polygons assigned to this node. Real number (x1) Can be used for collision detection.

Contents: Stores *display_size* number of <display> elements.

4.1.3.40 The <display> Element

Description: Holds the number of the polygon and the number of the material that will be rendered. Also holds the rendering priority that specifies rendering order

Output condition: Only output when the <node> stored in this element has a polygon (kind=mesh)

Output example:

```
display index="0" material="0" polygon="0" priority="1"/>
display index="1" material="1" polygon="2" priority="0"/>
</node>
```

Attributes:

Table 4-39 - The Attributes of the <display> Element

Attribute Name	Description
index	Index number. Integers 0 and larger (x1)
material	The material number to be used for drawing. Integer 0 or larger (x1) References <material> element
polygon	The polygon number to be drawn. Integer 0 or larger (x1) References <polygon> element
priority	Rendering priority determines from which <display> element (in order) rendering will be performed. (0 ≤ integer ≤ 255) x 1 Using this value, it is possible to control rendering order when displaying translucent polygons or decal polygons. If the number is 1 or more, rendering will be carried out in order from the <display> element with the smallest value. If the number is 0, no rendering order is specified. Rendering timing will depend on the rendering routine. If there are multiple identical values, the rendering routine will determine which to render first. The rendering routine determines whether rendering order will be controlled only within each node or for the model overall. For details see "Supplemental Information for imd Files", "Polygon Rendering Priority".

Contents: None

4.1.3.41 The <output_info> Element

Description: Stores the information that is output when the file is output from the 3D CG tool

Output condition: One of these is always output

Output example:

```
<output_info vertex_size="962" polygon_size="629"
triangle_size="620" quad_size="9"/>
```

Attributes:**Table 4-40 - The Attributes of the <output_info> Element**

Attribute Name	Description
vertex_size	Total number of vertices to be output by NITRO Geometry Engine. <i>Integer 0 or larger (x1)</i>
polygon_size	Total number of polygons (number of triangular polygons + number of quadrilateral polygons). <i>Integer 0 or larger (x1)</i>
triangle_size	Number of triangular polygons. <i>Integer 0 or larger (x1)</i>
quad_size	Number of quadrilateral polygons. <i>Integer 0 or larger (x1)</i>

Contents: None**4.1.3.42 The <ex_nns_3dme> Element****Description:** Stores the information specific to the 3D Material Editor.**Output condition:** Output when a file is output from the 3D Material Editor.**Output example:**

```

<ex_nns_3dme>
  <ex_model_info
    scene_name="test_scene"
    ... (omitted) ...
  />
  ... (omitted) ...
  <ex_***>
  ... (omitted) ...
  </ex_***>
</ex_nns_3dme>

```

Attributes: None**Contents:** Stores multiple child elements (Within the <ex_nns_3dme> element, all names of child elements begin with ex).

However, all child elements and attributes other than those stored under the <ex_nns_3dme> element are used by the 3D Material Editor, so the information needed to display 3D models on NITRO is not stored.

Note: The child elements and attributes stored in the <ex_nns_3dme> element are not described in this manual because they are not information related to displaying 3D models on NITRO.

4.1.4 Supplemental Information for imd Files

- The structure of nodes

The number of nodes that are output will differ, depending on the "node compression" setting at the time of output from the 3D CG tool.

To play ica, iva, or other animation correctly, the files must be output with the same node structure and the same node compression setting as the imd file.

- Relationship of Polygon to Material

The index number stored in the `<display>` element indicates which `<material>` is used to render the polygon stored in the `<polygon>` element.

- Character length of names

Single-byte alphanumeric characters must be used for the names of nodes, materials, texture images, texture palettes, and polygon groups in the imd file. There is no restriction on the character lengths of the names. However, if you are using the Nintendo NITRO-System library, then these names must be no longer than 16 characters in size.

- File names for textures

The texture image file name is stored in the texture image name. Even if the same texture image file is referenced by multiple materials in 3D CG Tools, it is stored as one piece of image data in the imd file. However, even if the file name is the same, if it uses a texture image file with a different path, there is the possibility that multiple instances of the texture image data with the same name will be created and the texture will not be able to be displayed normally.

- Texture Format

The texture formats below are supported:

- 4-color palette texture
- 16-color palette texture
- 256-color palette texture
- 4x4 texel compressed texture
- A3I5 translucent texture
- A5I3 translucent texture
- Direct texture

- Vertex color

1. Displaying with the light computation set to Off

If vertex color is configured for polygons and the vertex color is to be displayed on NITRO without performing the light computation, be sure to set the light Off for the corresponding material on the 3D CG tool. In this case, the `<clr>` element corresponds to the NITRO geometry command "Color."

2. Using vertex color and light computation together

By handling the `<clr>` element as the "MaterialColor0" diffuse color value instead of as the NITRO "Color" command, each vertex can be set to a different color, and expressions for calculating light are also possible.

- Envelopes in NINTENDO NITRO-System

The process of associating multiple nodes (also called skeletons and bones) with the polygon mesh and changing its shape as the nodes move is called "skinning" or "envelope." (The name depends on the 3D tool that is being used.) In NINTENDO NITRO-System, we call this feature "envelope".

In NINTENDO NITRO-System, there are two types of envelopes: **fully weighted envelopes** and **weighted envelopes**.

- Fully weighted envelopes

A fully weighted envelope is an envelope in which each vertex is fully weighted toward a single node. If all the vertices in a polygon mesh use fully weighted envelopes, the model is called a fully weighted envelope model.

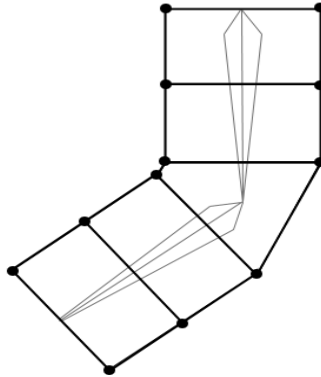


Figure 4-1 - An Example of a Fully Weighted Envelope Model

- Weighted envelopes

A weighted envelope is an envelope in which each vertex is associated with two or more nodes and the total weighting toward these nodes is 100%. If at least one vertex of a polygon mesh is a weighted envelope vertex, then the model is called a weighted envelope model.

A weighted envelope model allows you to create smooth curved surfaces by increasing the number of nodes associated with each vertex and by increasing the variety of weighting values. However, this increase will result in a greater amount of calculation during rendering.

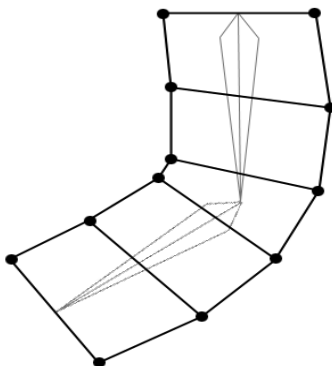


Figure 4-2 - An Example of a Weighted Envelope Model

Note: For information on support for weighted envelope models in the G3D library provided by NINTENDO NITRO-System, see the G3D Library Release Notes.

- Vertex Position Coordinates Transformation Matrix

The Vertex Position Coordinates Transformation Matrix references indices in the order of the `<mtx>` `<mtx_list>` `<matrix>` elements.

The imd file supports the calculation method that writes to the matrix stack with the Store command, which performs matrix calculations on the CPU. It also supports the matrix calculation method that uses the geometry commands PushMatrix and PopMatrix. When rendering the entire model, a system is used where a `<mtx_list>` element is provided and a 31-level matrix stack is set for each `<mtx_prim>`. This is done to support cases that exceed the 31-level matrix stack limitation of NITRO.

For vertices without a weighted envelope set (those that are dependent upon a matrix), the `mtx_weight` attribute of the referenced `<matrix>` element is set to 1. This type of vertex is displayed by using the node in the position indicated by the `(node_idx)th` attribute of the `<matrix>` element.

For vertices with a weighted envelope set (those that are weighted for multiple matrices), the `mtx_weight` attribute of the referenced `<matrix>` element is set to the number of nodes that have been weighted (2 or greater). This type of vertex references the `<weight>` and `<node_idx>` elements from `<matrix>`. It performs envelope calculations using the `(mtx_weight)` number from the `(envelope_head)` position of each `<weight>` and `<node_idx>` element.

For example, when `envelope_head = 2` and `mtx_weight = 2`:

```
<weight size="6">
50 50 40 60 40 60
</weight>
<node_idx size="6">
8 9 8 9 10 11
</node_idx>
```

In this case, the weighted envelope calculation is performed with a weight of 40% assigned to the index number 8 node and 60% assigned to the index number 9 node.

The vertex position coordinates of the envelope model are output as local coordinates relative to the node if the vertex is weighted 100% to a single node (a fully weighted envelope) and as global coordinates if the vertex's weight is distributed among multiple nodes (a weighted envelope).

Similarly, normal vectors are output as direction vectors in local space for fully weighted envelopes and as direction vectors in global space for weighted envelopes.

- Precision of Values

The vertex position value, normal vector value, and texture coordinate value output by the imd file are each calculated to the fixed decimal precision of their corresponding NITRO geometry command.

- Polygon Rendering Priority

When multiple materials are associated with a single mesh model (equivalent to the `<node>` element in the imd file) in a 3DCG tool, the multiple polygons that constitute the mesh model are divided into multiple polygon groups (equivalent to the `<polygon>` element in the imd file). In such a case, you can control the order in which polygon groups are rendered with respect to each material in the 3DCG tool by setting rendering priority (equivalent to the `priority` attribute of the `<display>` element in the imd file). Rendering priority is used when transparent polygons are displayed overlapping each other or when decal polygons are displayed.

Note: The "rendering order" mentioned here is the order in which render commands are sent to the NITRO geometry engine. According to the NITRO hardware specification, translucent polygons are always rendered after opaque polygons. Therefore, even if you set priority so that translucent polygons are rendered before opaque polygons, rendering in NITRO will always start from opaque polygons.

If you want to manage rendering order for a polygon group, order it by setting a rendering priority of at least 1 (the lower the value, the earlier the rendering).

If there is no need to specify rendering order for a polygon group, set the rendering priority to 0. The rendering routine will determine the rendering timing for polygon groups that have a rendering priority of 0. If there are multiple polygon groups with the same rendering priority, the rendering routine will determine which polygon group to render from.

`<rendering priority>`

0: No order specified (rendering timing is indeterminate).

1 or more: Render in order from the one with the smallest value.

The rendering routine determines whether to control render order inside each node, or in the overall model.

For example, in the case of the following example, rendering order changes according to whether control is done inside each node or for the model overall.

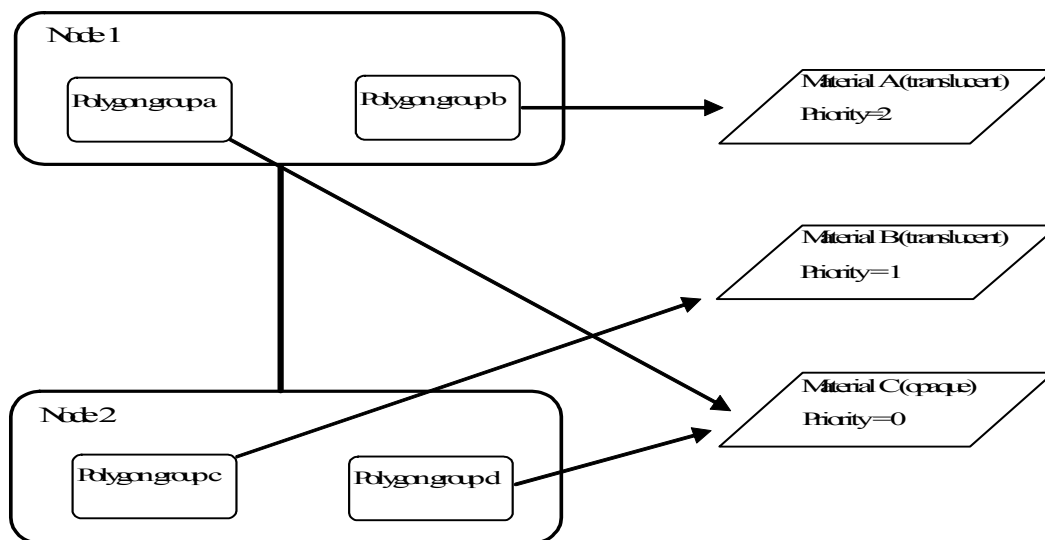


Figure 4-3 - Polygon Rendering Priority Example

- Controlling rendering order in each node (equivalent to the <node> element in the imd file)

Uses a rendering routine that employs generic Push/Pop matrix calculation. When rendering in order from the parent node, it renders the polygon groups on a node basis. Therefore the overall model sends render commands in this order:

"(polygon group **a**) > **b** > (**a**) > (**d**) > **c** (**d**)" (either **a** or **d**)

- Controlling order over the entire model (imd file)

In the case of a rendering routine that first does the required matrix calculations and then collectively sends render commands, there is no need for the rendering order to depend on the parent/child relationship. Therefore, it is possible to render all of the polygon groups in the model according to the rendering priority. In a node configuration like that shown above, render commands would be sent as shown below.

"(polygon group **a,d**) → **c** → (**a,d**) → **b** → (**a,d**)" (either **a** or **d**)

Since the rendering priority for polygon groups **a** and **d** is 0, the rendering routine will determine when to render each.

Note: The rendering routine in the G3D library that is supplied with the NINTENDO NITRO-System controls rendering order using the latter: overall model rendering order.

- The material compression feature and polygon groups

If within the same node two different materials have the same content and the same rendering priority, when you use the material compression feature in the intermediate file export plug-in, the material and the polygon groups will each be combined into one.

If two materials are the same but their rendering priorities are different, the materials will be combined into one, but the polygon groups will not because their rendering priorities are different.

If different nodes contain the same materials and the same rendering priorities, the polygon groups will be combined into one, only if Unite and Combine Polygon is specified when culling nodes.

- About Environmental and Projection Mapping

By setting the texture coordinate conversion mode to either Normal or Vertex in NITRO, displays known as "environmental mapping" and "projection mapping" can be performed.

With the intermediate file format, the attribute `tex_effect_mtx` has been prepared for use inside the `<material>` element in order to designate from where and in what way the mapping will be performed when environmentally or projectionally mapping.

Note: There is no "properly defined" way of using `tex_effext_mtx` with the intermediate file format. The usage method should be determined on the render routine side.

The G3D libraries provided by NINTENDO NITRO-System are used in the following ways:

- Example 1) Environmental Mapping

Environmental mapping can be expressed on NITRO by setting the texture coordinate conversion mode to a Normal source, setting the results of the matrix calculation below to the texture matrix, and sending the normal vector with a geometry command.

(The matrix that performs conversion to a view coordinate-type normal vector) *
`(tex_effect_mtx)` *
 (The matrix that lines up the center and size of the mapping (a constant)) *
 (The texture scale and rotate being designated with the material)

It is possible to have expressions for projecting the texture from various fixed directions by configuring the rotation matrix and the scaling matrix in `tex_effect_mtx`.

- Example 2) Projection Mapping

Projection mapping can be expressed on NITRO by setting the texture coordinate conversion mode to a Vertex source, setting the results of the matrix calculation below to the texture matrix, and sending the vertex coordinates with a geometry command.

(The matrix that performs conversion to world coordinate-type vertex coordinates) *
`(tex_effect_mtx)` *
 (The matrix that lines up the center and size of the mapping (a constant)) *

It is possible to have expressions for illuminating the texture from various fixed directions by configuring the rotation matrix and the scaling matrix in `tex_effect_mtx`.

Special mapping expressions other than those described above can be achieved by reworking the texture matrix.

Note: Regarding the attribute `tex_gen_st` in the `<material>` element, the information used to express the texture environment and projection mapping was included in the intermediate file format up to version 1.5.0, but since this duplicates the contents of the attribute `tex_effect_mtx` added from version 1.6.0, there is no further need to use that old information and it has effectively been discarded.

5 Intermediate Files for Animation

5.1 The Data Formats of Animation Files

5.1.1 Types of Data Formats

The Intermediate files for NITRO support five types of animation: character animation, visibility animation, material color animation, texture pattern animation, and texture SRT animation.

There are two formats for animation data: **Frame format** and **FV Key format**. Each animation file uses an optimized data format.

Table 5-1 - Animation Data Formats

Data Format	Special Features	NITRO Intermediate File
Frame format	<p>In this format, the animation data on the 3D CG tool is output in units of frames.</p> <p>The volume of data increases as the number of frames increase, but the size can be reduced by culling data frames at fixed intervals. (Culling reduces precision, but errors can be minimized by linearly interpolating between the data frames.)</p>	ica ima ita
FV key format	<p>In this format, a set of (<i>Frame</i>, <i>Value</i>) is treated as one set of key data, and the applicable frame is searched for among multiple sets of key data to use its <i>Value</i> value.</p> <p>The volume of data is not related to the number of frames; instead, it is proportional to the extent of variation in values.</p> <p>Note: This differs from the Hermite interpolation key data used by regular 3D CG tools, where the set is comprised of (<i>Frame</i>, <i>Value</i>, <i>Slope</i>).</p> <p>No decision has been made yet whether there will be support in the future for files output in Hermite format.</p>	itp iva

5.1.2 Frame Format

5.1.2.1 Frame Format Data Structure

In Frame format, data for all frames from the 3D CG tool are output. In order to reduce the overall volume of data, this format can accommodate a culling of frames at fixed intervals.

frame_step lets you select from among three levels of culling:

- When **frame_step** = 1, no culling is done and the same number of frames from the 3D CG tool are output.
- When **frame_step** = 2, data is output for every other frame, reducing the data to 1/2 its original size.
- When **frame_step** = 4, data is output for every fourth frame, reducing the data to 1/4 its original size.

A larger **frame_step** value reduces the quality of the animation, but this is the tradeoff for reducing the volume of data. (There is no **frame_step**= 3 because NITRO uses shifting not division.)

The output data consists of every n^{th} frame, where n is the multiple defined by **frame_step, and all frames that remain at the end.** The table illustrates the differing amounts of frame data that are output to the Frame file depending on the **frame_step** value when the 3D CG tool has created 16 frames of animation.

Table 5-2 - Frames Output in Frame Format

Frame	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
When frame_step = 1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
When frame_step = 2	0		2		4		6		8		10		12		14	15
When frame_step = 4	0				4				8				12	13	14	15

Use linear interpolation on the prior and subsequent frame data to get values for the data that have been culled, as well as for fractional frames calculated when the playback speed is changed.

When outputting the Intermediate file from the 3D CG tool, select a value of 1, 2, 4 or auto for `frame_step`. When 1, 2, or 4 is selected, data is output for the entire animation curve in accordance with the specified `frame_step` value. When "auto" is selected, the optimal `frame_step` value is calculated automatically from within the plug-in, determining the best value in units of animation curves that fits within a specially set range of tolerable error.

5.1.2.2 Frame Format Playback Algorithm

The algorithm for playback of Frame format data can be described as the following C language function. To make the example easier to understand, the playback of fractional frames and the process of interpolating from End frame to Start frame have been left out. Assume that a 4096-step sine table is used for the Rotate data.

Code 1 - Sample of the Frame format playback function (in C source format)

```

const int frame_size;          // Total number of frames
int      now_frame;            // Current frame
int getLinearInterpValue(
    const int frame_step,      // Extent of culling
    const int data_size,       // Data size
    const int* value,          // Actual data array
    const bool angle_flag)     // Flag indicating if Rotate element or not
{
    int shift, ival, iframe_last_interp;
    int val0, val1, weight0, weight1, result;

    if(data_size == 1)         // When data = 1
        return value[0];

    if(frame_step == 1)        // When frame_step = 1
        return value[now_frame];

    shift = (frame_step==2) ? 1 : 2; // Shift value
    ival = now_frame >> shift; // Position of actual data that corresponds to
                                // current frame

    // The frame that is the threshold between culled part and all-frames part -----
    iframe_last_interp = ((frame_size - 1) >> shift) << shift;

    // Directly return from actual data array when at or after the threshold frame -----
    if (now_frame >= iframe_last_interp)
        return value[ival + (now_frame - iframe_last_interp)];

    // When before the threshold frame, get with interpolation computation -----
    val0 = value[ival];
    val1 = value[ival + 1];
    weight1 = now_frame - (ival << shift);
    weight0 = frame_step - weight1;

    // When just at frame -----
    if (weight1 == 0)
        return val0;

    // Special process when target is a Rotate element -----
    else if( angle_flag && abs(val0 - val1) >= 2048){
        if(val0 > val1) val1 += 4096;
        else val0 += 4096;
        result = ( val0 * weight0 + val1 * weight1) >> shift );
        if(result > 4095) result -= 4096;
        return result;
    }

    // Regular interpolation process -----
    else {
        return ((val0 * weight0 + val1 * weight1) >> shift);
    }
}

```

5.1.2.3 The Process of Interpolating From End Frame to Start Frame

To slowly interpolate and play looped animation requires a smooth connection from the last frame to the first frame.

The Frame-format Intermediate file has a flag that indicates whether interpolation is conducted from the End frame to the Start frame. Assume that you have an animation consisting of 100 frames of data. At playback time, you can specify playback of 0 to 99.9999 frames (the 100th frame = the 0th frame). When the animation playback speed is set in units of frames, then play proceeds as 0, 1, 2 ... 98, 99. However, if the playback speed is set to units of half frames (0.5 frame), then play proceeds as 0.0, 0.5, 1.0, 1.5 ... 98.5, 99.0, 99.5. In the latter case, playback can exceed the final frame (99.0). Suppose that you have calculated the 99.5th frame; the state that is returned will be the same as that for the 99th frame (the final frame) if interpolation is not conducted from the End frame to the Start frame. To interpolate from the End frame to the Start frame, you must interpolate the 99th and 0th frames.

5.1.3 FV Key Format

5.1.3.1 Key Format Data Structure

In this format, a set of (*Frame*, *Value*) is treated as one set of key data, and the applicable frame is searched for among multiple sets of key data to use its *Value* value.

If the 3D CG tool creates an animation with 16 frames of animation data, then the Intermediate file is output with the data structure shown in the table below.

Table 5-3 - Data on the 3D CG Tool and FV Key Data

Data on the 3D CG Tool

Frame	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Value	0	0	0	0	1	1	1	2	2	2	2	1	1	1	0	0

FV Key Data

Frame	0	4	7	11	14
Value	0	1	2	1	0

5.1.3.2 FV Key Format Playback Algorithm

The Frame data is searched to find the frame that is the same as the present frame or the immediately prior frame, and the corresponding *Value* value is used. In the FV Key format, interpolation as well as the interpolation process from End frame to Start frame are not conducted.

Code 2 - Sample of the FV Key format function (in C source format)

```

int getFVkeyValue(const int* frames,// Frame number string
                  const int* values,// Actual data array
                  const int data_size,// Number of data
                  const int iframe)// Current frame
{
// When there is only one set of data -----
    if(data_size==1)
        return values[0];

// When there are 2 or more sets of data -----
    else {
        int LeftKey=0, RightKey=data_size-1, CenterKey;

        // When current frame is before first Key -----
        if(iframe <= frames[LeftKey])
            return values[LeftKey];

        // When current frame is after last Key -----
        if(iframe >= frames[RightKey])
            return values[RightKey];

        // When between any two Keys -----
        while(1){
            if((RightKey-LeftKey)==1 || RightKey==LeftKey){
                if(iframe < RightKey)
                    return values[LeftKey];
                else
                    return values[RightKey];
            }
            CenterKey = (LeftKey + RightKey)/2;
            if(iframe <= frames[CenterKey])
                RightKey = CenterKey;
            else if(iframe >= frames[CenterKey])
                LeftKey = CenterKey;
        }
    }
}

```

5.2 Character Animation Files (ica)

5.2.1 Overview of ica Files

The Character Animation file (ica: Intermediate Character Animation file) stores the data for moving the model. Specifically, it holds the data that is output when the matrices held in every node are multiplied by the Scale, Rotate, and Translate matrices. Character animation data is output in Frame format.

No decision has been made yet whether there will be support for Hermite-interpolated Key format output that many 3D CG tools use.

5.2.2 The Basic Element Structure of ica Files

<ica>	...Root
<head>	...Information relating to overall document
... (Omitted) ...	
/head>	
<body>	...The data itself
<node_anm_info />	...Animation settings information
<node_scale_data />	...Scale data
<node_rotate_data />	...Rotate data
<node_translate_data />	...Translate data
<node_anm_array>	...Node array
<node_anm />	...Animation reference information
<scale_x />	...Scale X axis reference information
<scale_y />	...Scale Y axis reference information
<scale_z />	...Scale Z axis reference information
<rotate_x />	...Rotate X axis reference information
<rotate_y />	...Rotate Y axis reference information
<rotate_z />	...Rotate Z axis reference information
<translate_x />	...Translate X axis reference information
<translate_y />	...Translate Y axis reference information
<translate_z />	...Translate Z axis reference information
</node_anm />	
</node_anm_array>	
/body>	
</ica>	

5.2.3 Explanation of ica Elements and Attributes

5.2.3.1 The <node_anm_info> Element

Description: Stores information relating to the overall character animation

Output condition: Always output

Output example:

```

<node_anm_info
    frame_size="90"
    scaling_rule="standard"
    magnify="1.000000"
    tool_start_frame="1"
    tool_end_frame="90"
    interpolation="linear"
interp_end_to_start="off"
    compress_node="none" node_size="31 31"
    frame_step_mode="auto"
    tolerance_scale="0.100000"
    tolerance_rotate="0.100000"
    tolerance_translate="0.100000"
/>

```

Attributes:**Table 5-4 - The Attributes of the <node_anm_info> Element**

Attribute Name	Description
frame_size	The number of frames. Integer 1 or larger (x1)
scaling_rule	<p>Scaling calculation method. Node structure for any of these character strings (standard, maya, si3d) is a: parent, b: child, c: descendant.</p> <p>standard: Standard matrix calculation: (Sc * Rc * Tc * Sb * Rb * Tb * Sa * Ra * Ta)</p> <p>maya: Calculation that considers Segment Scale Compensate in maya.</p> <p>si3d: SOFTIMAGE matrix calculation method</p> <p>For information on each calculation method, see Node Matrix Computation Methods.</p>
magnify	<p>The scale factor applied to the entire model when outputting the imd from 3D CG tools. Real number value x1</p> <p>This information is for the confirmation of settings during intermediate file output.</p> <p>This reflects the vertex position coordinates, translate values, etc., in the intermediate file, so there is no need to use this value in NITRO.</p>
tool_start_frame	<p>The StartFrame specified when outputting file from 3D CG tool. Integer (x1)</p> <p>(To check the ica output state)</p>
tool_end_frame	<p>The EndFrame specified when outputting file from 3D CG tool. Integer (x1)</p> <p>(To check the ica output state)</p>
interpolation	<p>Specifies the method to use when values from in-between frames are obtained when playing animation. Character string [frame / linear]</p> <p>frame: Discard in-between frames and use integer frames to play.</p> <p>linear: Do not discard in-between frames—use them as they are. When playing, use linear interpolation to obtain in-between frame values.</p> <p>Also refer to "Cautions When Playing Using the G3D Library"</p>
interp_end_to_start Output only when interpolation is <i>linear</i>	<p>Flag indicating whether to perform linear interpolation between the final frame and the first frame. Character string [off / on]</p> <p>off: Do not change beyond final frame</p> <p>on: Perform linear interpolation</p>

Attribute Name	Description
compress_node	<p>The compression state of the output nodes.</p> <p>Character string [none / cull / merge / unite / unite_combine]</p> <p>none: Output the same node structure as on the tool.</p> <p>cull: Output after culling nodes not needed for model display.</p> <p>merge: In addition to culling, merge all possible node matrices and then output.</p> <p>unite: Unite all nodes into a single node. (Primarily used for topographic data)</p> <p>unite_combine: In addition to the unite process, also unite all <polygon> elements that share the same <material> element.</p> <p>(For checking the ica output state)</p>
node_size	<p>The number of nodes when uncompressed and compressed. Integer 1 or larger (x2)</p> <p>When compress_node = none, the number of uncompressed nodes is listed twice.</p>
frame_step_mode	<p>Data culling specification. Character string [1 / 2 / 4 / auto]</p> <p>1: Output all frames that are on the 3D CG tool.</p> <p>2: Output every other frame (= around 1/2 animation data size)</p> <p>4: Output every fourth frame (= around 1/4 animation data size)</p> <p>auto: Automatically determine the optimal extent of culling for every animation curve.</p>
tolerance_scale	Tolerable error when culling Scale data. Real number 0.0 or larger (x1)
tolerance_rotate	Tolerable error when culling Rotate data. Real number 0.0 or larger (x1)
tolerance_translate	Tolerable error when culling Translate data. Real number 0.0 or larger (x1)

Contents: None

5.2.3.2 The <node_scale_data> Element

Description: Stores Scale animation data

Output condition: Always output

Output example:

```
<node_scale_data size="3">
    1.000000 1.500000 2.300000
</node_scale_data>
```

Attributes:

Table 5-5 - The Attributes of the <node_scale_data> Element

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores size number of real numbers (Scale values)

5.2.3.3 The <node_rotate_data> Element

Description: Stores Rotate animation data

Output condition: Always output

Output example:

```
<node_rotate_data size="1722">
  0.000000 124.377441 0.366699 0.363525 0.354492
  ... (Omitted) ...
0.135742 0.162354
</node_rotate_data>
```

Attributes:**Table 5-6 - The Attributes of the <node_rotate_data> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of real numbers (Rotate values: degrees)

5.2.3.4 The <node_translate_data> Element

Description: Stores Translate animation data

Output condition: Always output

Output example:

```
<node_translate_data size="110">
0.027100 0.037598 0.049072 0.061035 0.073730
  ... (Omitted) ...
-0.003662 -0.033691 4.086426 -0.246338 -1.791992
</node_translate_data>
```

Attributes:**Table 5-7 - The Attributes of the <node_translate_data> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of real numbers (Translate values)

5.2.3.5 The <node_anm_array> Element

Description: Stores the animation's reference information in an array

Output condition: Always output

Output example:

```
<node_anm_array size="2">
  node_anm index="0">
    ... (Omitted) ...
  </node_anm>
  <node_anm index="1">
    ... (Omitted) ...
  </node_anm>
</node_anm_array>
```

Attributes:**Table 5-8 - The Attributes of the <node_anm_array> Element**

Attribute Name	Description
size	The number of <node_anm> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <node_anm> elements.

5.2.3.6 The <node_anm> Element

Description: Stores the animation's reference information

Output condition: Always output one or more

Output example:

```
<node_anm_array size="2">
<node_anm index="0">
  <scale_x frame_step="1" data_size="1" data_head="0"/>
  <scale_y frame_step="1" data_size="1" data_head="0"/>
  <scale_z frame_step="1" data_size="1" data_head="0"/>
  <rotate_x frame_step="1" data_size="1" data_head="0"/>
  <rotate_y frame_step="1" data_size="1" data_head="0"/>
  <rotate_z frame_step="1" data_size="1" data_head="0"/>
  <translate_x frame_step="1" data_size="90" data_head="0"/>
  <translate_y frame_step="1" data_size="1" data_head="90"/>
  <translate_z frame_step="1" data_size="1" data_head="91"/>
</node_anm>
<node_anm index="1">
  ... (Omitted) ...
</node_anm>
</node_anm_array>
```

Attributes:**Table 5-9 - The Attributes of the <node_anm> Element**

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)

Contents: Stores one of each of these elements in this order: <scale_x>, <scale_y>, <scale_z>, <rotate_x>, <rotate_y>, <rotate_z>, <translate_x>, <translate_y>, and <translate_z>.

5.2.3.7 The <scale_x>, <scale_y>, and <scale_z> Elements

Description: Stores the Scale animation's reference information

Output condition: Always output

Output example:

```

<node_anm index="0">
<scale_x frame_step="1" data_size="1" data_head="0"/>
<scale_y frame_step="1" data_size="1" data_head="0"/>
<scale_z frame_step="2" data_size="45" data_head="0"/>
<rotate_x frame_step="1" data_size="1" data_head="0"/>
<rotate_y frame_step="1" data_size="1" data_head="1"/>
<rotate_z frame_step="1" data_size="1" data_head="2"/>
<translate_x frame_step="1" data_size="90" data_head="0"/>
<translate_y frame_step="1" data_size="1" data_head="90"/>
<translate_z frame_step="1" data_size="1" data_head="91"/>
</node_anm>

```

Attributes:**Table 5-10 - The Attributes of the <scale_x>, <scale_y>, <scale_z> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output data for all frames 2: Output every other frame (= around 1/2 animation data size) 4: Output every fourth frame (= around 1/4 animation data size)
data_size	The data size of the animation. Integer 1 or larger (x1) References <node_scale_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <node_scale_data>

Contents: None**5.2.3.8 The <rotate_x>, <rotate_y>, and <rotate_z> Elements****Description:** Stores the Rotate animation's reference information**Output condition:** Always output**Output example:**

```

<node_anm index="0">
<scale_x frame_step="1" data_size="1" data_head="0"/>
<scale_y frame_step="1" data_size="1" data_head="0"/>
<scale_z frame_step="2" data_size="45" data_head="0"/>
<rotate_x frame_step="1" data_size="1" data_head="0"/>
<rotate_y frame_step="1" data_size="1" data_head="1"/>
<rotate_z frame_step="1" data_size="1" data_head="2"/>
<translate_x frame_step="1" data_size="90" data_head="0"/>
<translate_y frame_step="1" data_size="1" data_head="90"/>
<translate_z frame_step="1" data_size="1" data_head="91"/>
</node_anm>

```

Attributes:**Table 5-11 - The Attributes of the <rotate_x>, <rotate_y>, <rotate_z> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output data for all frames 2: Output every other frame (= around 1/2 animation data size) 4: Output every fourth frame (= around 1/4 animation data size)
data_size	Amount of animation data. Integer 1 or larger (x1) References <node_rotate_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <node_rotate_data>

Contents: None**5.2.3.9 The <translate_x>, <translate_y>, and <translate_z> Elements****Description:** Stores the Translate animation's reference information**Output condition:** Always output**Output example:**

```
<node_anm index="0">
<scale_x frame_step="1" data_size="1" data_head="0"/>
<scale_y frame_step="1" data_size="1" data_head="0"/>
<scale_z frame_step="2" data_size="45" data_head="0"/>
<rotate_x frame_step="1" data_size="1" data_head="0"/>
<rotate_y frame_step="1" data_size="1" data_head="0"/>
<rotate_z frame_step="1" data_size="1" data_head="0"/>
<translate_x frame_step="1" data_size="90" data_head="0"/>
<translate_y frame_step="1" data_size="1" data_head="90"/>
<translate_z frame_step="1" data_size="1" data_head="91"/>
</node_anm>
```

Attributes:**Table 5-12 - The Attributes of the <translate_x>, <translate_y>, <translate_z> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output data for all frames 2: Output every other frame (= around 1/2 animation data size) 4: Output every fourth frame (= around 1/4 animation data size)
data_size	Amount of animation data. Integer 1 or larger (x1) References <node_translate_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <node_translate_data>

Contents: None

5.2.4 Supplemental Information for ica Files

- Relation to imd

To play back ica properly, an imd file with the same node structure is required.

imd and ica correspond to the imd `<node>` element and the ica `<node_anm>` element indices.

Even with the same model, the animation may not be reflected properly in the imd file if it has a different node structure. For example, this might occur because the node-compression setting was different when it was output from the 3D CG tool.

- Cautions When Playing Using the G3D Library

With the G3D library released in the current NITRO-System library it is possible to play in-between frames by setting interpolation to linear. However, compared to frame, interpolation this will slightly increase the CPU calculation process. If animation data does not require setting interpolation to linear, with respect to lightening the above-mentioned CPU processing load, be careful to output with interpolation set to frame.

5.3 Visibility Animation Files (iva)

5.3.1 Overview of iva Files

The Visibility Animation file (iva: Intermediate Visibility Animation file) holds data that has been output for animations for which visibility (show/hide) switches for every node. Visibility animation is output in FV Key format, in which "frame number" and "show/hide state" are paired as a single group.

5.3.2 The Basic Element Structure of iva Files

<code><iva></code>	... Root
<code><head></code>	... Information relating to overall document
... (Omitted) ...	
<code>/head></code>	
<code><body></code>	... The data itself
<code><visibility_info /></code>	... Animation settings information
<code><visibility_data></code>	... Animation data
<code><frame_idx /></code>	... Frame data
<code><visibility /></code>	... Display show/hide data
<code>/visibility_data></code>	
<code><visibility_anm_array></code>	... Animation array
<code><visibility_anm /></code>	... Animation reference information
<code>/visibility_anm_array></code>	
<code>/body></code>	
<code></iva></code>	

5.3.3 Explanation of iva Elements and Attributes

5.3.3.1 The `<visibility_info>` Element

Description: Stores overall information relating to visibility animation

Output condition: Always output

Output example:

```
<visibility_info  
  frame_size="55"  
  tool_start_frame="1"  
  tool_end_frame="55"  
  compress_node="none" node_size="10 10"  
>
```

Attributes:**Table 5-13 - The Attributes of the <visibility_info> Element**

Attribute Name	Description
frame_size	The number of frames. Integer 1 or larger (x1)
tool_start_frame	The StartFrame specified when outputting a file from the 3D CG tool. Integer (x1) (For checking the iva output state)
tool_end_frame	The EndFrame specified when outputting a file from the 3D CG tool. Integer (x1) (For checking the iva output state)
compress_node	The compression state of the output node Character string [none / cull / merge / unite / unite_combine] none: Outputs the same node structure as on the tool. cull: Outputs after culling nodes not needed for model display. merge: In addition to the cull process, merge all node matrices that can be merged and then output. unite: Unite all nodes into a single node. (This is mainly used for topographic data) unite_combine: In addition to the unite process, also unites all <polygon> elements that share the same <material> element. (For checking the iva output state)
node_size	The number of nodes when uncompressed and after compression. Integer 1 or larger (x2) When <code>compress_node=none</code> , the number of uncompressed nodes is listed twice. (For checking the iva output state)

Contents: None**5.3.3.2 The <visibility_data> Element****Description:** Stores the frame data and visibility data**Output condition:** Always output

Output example:

```
<visibility_data>
  <frame_idx size="28">
    0 0 4 9 0 9 14 0 14 19
    0 19 24 0 24 29 0 29 34 0
    34 39 0 39 44 0 44 49
  </frame_idx>
  <visibility size="28">
    1 1 0 1 1 0 1 1 0 1
    1 0 1 1 0 1 1 0 1 1
    0 1 1 0 1 1 0 1
  </visibility>
</visibility_data>
```

Attributes: None**Contents:** Stores one `<frame_idx>` element and one `<visibility>` element, in that order**5.3.3.3 The `<frame_idx>` Element****Description:** Stores the frame dataThe number of data sets and the order matches that of the `<visibility>` element**Output condition:** Always output**Output example:**

```
<visibility_data>
  frame_idx size="28">
    0 0 4 9 0 9 14 0 14 19
    0 19 24 0 24 29 0 29 34 0
    34 39 0 39 44 0 44 49
  </frame_idx>
  <visibility size="28">
    1 1 0 1 1 0 1 1 0 1
    1 0 1 1 0 1 1 0 1 1
    0 1 1 0 1 1 0 1
  </visibility>
</visibility_data>
```

Attributes:**Table 5-14 - The Attributes of the `<frame_idx>` Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of integers, 0 or larger (frame number)**5.3.3.4 The `<visibility>` Element****Description:** Stores the visibility data (The show/hide state. 0 = hide, 1 = show)The number of data sets and the order matches that of the `<frame_idx>` element**Output condition:** Always output

Output example:

```
<visibility_data>
  <frame_idx size="28">
    0 0 4 9 0 9 14 0 14 19
    0 19 24 0 24 29 0 29 34 0
    34 39 0 39 44 0 44 49
  </frame_idx>
  visibility size="28">
    1 1 0 1 1 0 1 1 0 1
    1 0 1 1 0 1 1 0 1 1
    0 1 1 0 1 1 0 1
  </visibility>
</visibility_data>
```

Attributes:**Table 5-15 - The Attributes of the <visibility> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of the integers 0 (hide) or 1 (show).

5.3.3.5 The <visibility_anm_array> Element

Description: Stores visibility animation reference information for only the number of nodes

Output condition: Always output

Output example:

```
<visibility_anm_array size="10">
  <visibility_anm_index="0" data_size="1" data_head="0"/>
  ... (Omitted) ...
  <visibility_anm index="9" data_size="3" data_head="25"/>
</visibility_anm_array>
```

Attributes:**Table 5-16 - The Attributes of the <visibility_anm_array> Element**

Attribute Name	Description
size	The amount of <visibility_anm> data stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <visibility_anm> elements.

Note: The order and the number of <visibility_anm> elements correspond to that in the <node> element of the corresponding imd file.

5.3.3.6 The <visibility_anm> Element

Description: Stores visibility animation reference information for each node

Output condition: Always output

Output example:

```

<visibility_anm_array size="10">
  visibility_anm index="0" data_size="1" data_head="0"/>
  <visibility_anm index="1" data_size="3" data_head="1"/>
  <visibility_anm index="2" data_size="3" data_head="4"/>
  <visibility_anm index="3" data_size="3" data_head="7"/>
  <visibility_anm index="4" data_size="3" data_head="10"/>
  <visibility_anm index="5" data_size="3" data_head="13"/>
  <visibility_anm index="6" data_size="3" data_head="16"/>
  <visibility_anm index="7" data_size="3" data_head="19"/>
  <visibility_anm index="8" data_size="3" data_head="22"/>
  <visibility_anm index="9" data_size="3" data_head="25"/>
</visibility_anm_array>

```

Attributes:**Table 5-17 - The Attributes of the <visibility_anm> Element**

Attribute Name	Description
index	Index number. Integer 0 or larger (x1) Corresponds to the imd node number
data_size	Size of animation data. Integer 1 or larger (x1) References the <frame_idx> and <visibility> elements
data_head	Location of start of data. Integer 0 or larger (x1) References the <frame_idx> and <visibility> elements

Contents: None**5.3.4 Supplemental Information for iva Files**

- Relation to imd

To play back iva properly, an imd file with the same node structure is required.

imd and iva correspond to the imd <node> element and the iva <visibility_anm> element indices.

Even with the same model, the animation may not be reflected correctly in the imd file if it has a different node structure because, for example, the node-compression setting was different when it was output from the 3D CG tool.

5.4 Material Color Animation Files (ima)

5.4.1 Overview of ima Files

The Material Color Animation file (ima: Intermediate Material Color Animation file) stores the diffuse color, ambient color, specular color, and emission color for each material, as well as animation data to change polygon alpha. Material color animation data is output in Frame format.

No decision has been made yet whether there will be support for Hermite-interpolated Key format output that many 3D CG tools use.

5.4.2 The Basic Element Structure of ima Files

```
<ima> ... Root
  <head>... Information relating to overall document
    ... (Omitted) ...
  /head>
  <body>... The data itself
    <mat_color_info />... Animation settings information
    <mat_color_data />... Color data
    <mat_color_anm_array>... Animation array
    <mat_color_anm>... Animation reference information
      <diffuse_r>... Diffuse red component's reference information
      <diffuse_g>... Diffuse green component's reference information
      <diffuse_b>... Diffuse blue component's reference information
      <ambient_r>... Ambient red component's reference information
      <ambient_g>... Ambient green component's reference information
      <ambient_b>... Ambient blue component's reference information
      <specular_r>... Specular red component's reference information
      <specular_g>... Specular green component's reference information
      <specular_b>... Specular blue component's reference information
      <emission_r>... Emission red component's reference information
      <emission_g>... Emission green component's reference information
      <emission_b>... Emission blue component's reference information
      <polygon_alpha>... Polygon alpha's reference information
    /mat_color_anm>
  /mat_color_anm_array>
/body>
/</ima>
```

5.4.3 Explanation of ima Elements and Attributes

5.4.3.1 The <mat_color_info> Element

Description: Stores overall information relating to material color animation

Output condition: Always output

Output example:

```
<mat_color_info
  frame_size="100"
  tool_start_frame="1"
  tool_end_frame="100"
  interpolation="linear"
  interp_end_to_start="off"
  compress_material="on" material_size="1 1"
  frame_step_mode="auto"
  tolerance_color="1"
/>
```

Attributes:**Table 5-18 - The Attributes of the <mat_color_info> Element**

Attribute Name	Description
frame_size	The number of frames. Integer 1 or larger (x1)
tool_start_frame	The StartFrame specified when outputting file from 3D CG tool. Integer (x1) (For checking the ima output state)
tool_end_frame	The EndFrame specified when outputting file from 3D CG tool. Integer (x1) (For checking the ima output state)
interpolation	Specifies the method to use when values from in-between frames are obtained when playing animation. Character string [frame / linear] frame: Discard in-between frames and use integer frames to play. linear: Do not discard in-between frames—use them as they are. When playing, use linear interpolation to obtain in-between frame values. Also refer to "Cautions When Playing Using the G3D Library"
interp_end_to_start Output only when interpolation is <i>linear</i>	Flag indicating whether to perform linear interpolation between the final frame and the first frame. Character string [off / on] off: Do not change beyond final frame on: Perform linear interpolation
compress_material	The material's compression state. Character string [off / on] off: Output the same number of materials as on tool. on: If materials with exact same settings exist, only output the one that comes first alphabetically. (For checking the ima output state)
material_size	The number of materials when uncompressed and when compressed. Integer 0 or larger (x2) When compress_material=off, the number of uncompressed materials is listed twice. (For checking the ima output state)
frame_step_mode	Data culling specification. Character string [1 / 2 / 4 / auto] 1: Output all frames that are on the 3D CG tool. 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size) auto: Automatically calculate optimal culling for every animation curve.
tolerance_color	Tolerable error when culling color data. Integer between 0 and 31 (x1)

Contents: None**5.4.3.2 The <mat_color_data> Element****Description:** Stores color data**Output condition:** Output only when material color animation has been configured

Output example:

```
<mat_color_data size="85">
  25 23 21 18 16 14 11 9 7 5
  3 2 1 0 0 0 0 0 0
  0 0 0 0 0 0 0 0 0 2
  5 7 10 13 15 18 20 22 23 24
  25 25 24 23 21 19 17 14 12 9
  7 4 2 1 1 0 1 0 0 0
  0 0 0 0 0 0 0 0 0 1
  2 4 6 8 10 12 14 17 19 21
  23 24 24 25 31
</mat_color_data>
```

Attributes:**Table 5-19 - The Attributes of the <mat_color_data> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of integer values, where integer takes value between 0 and 31

Note: In accordance with the NITRO specifications, R, G, B, A shades are normalized and output as integers taking values between 0 and 31.

5.4.3.3 The <mat_color_anm_array> Element

Description: Stores reference information for the number of material color animations

Output condition: Outputs only when material color animation has been configured

Output example:

```
<mat_color_anm_array size="2">
  mat_color_anm index="0" material_name="lambert2">
    ... (Omitted) ...
  </mat_color_anm>
<mat_color_anm index="1" material_name="lambert3">
  ... (Omitted) ...
</mat_color_anm>
</mat_color_anm_array>
```

Attributes:**Table 5-20 - The Attributes of the <mat_color_anm_array> Element**

Attribute Name	Description
size	The number of <mat_color_anm> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <mat_color_anm> elements.

The <mat_color_anm> elements are output by material_name in alphanumerical order (a→z and 0→9).

5.4.3.4 The <mat_color_anm> Element

Description: Stores the material color animation's reference information

Output condition: Outputs only when material color animation has been configured

Output example:

```
<mat_color_anm_array size="1">
  mat_color_anm index="0" material_name="lambert2">
    diffuse_r frame_step="4" data_size="28" data_head="0"/>
    <diffuse_g frame_step="4" data_size="28" data_head="28"/>
    <diffuse_b frame_step="4" data_size="28" data_head="56"/>
    <ambient_r frame_step="1" data_size="1" data_head="13"/>
    <ambient_g frame_step="1" data_size="1" data_head="13"/>
    <ambient_b frame_step="1" data_size="1" data_head="13"/>
    <specular_r frame_step="1" data_size="1" data_head="13"/>
    <specular_g frame_step="1" data_size="1" data_head="13"/>
    <specular_b frame_step="1" data_size="1" data_head="13"/>
    <emission_r frame_step="1" data_size="1" data_head="13"/>
    <emission_g frame_step="1" data_size="1" data_head="13"/>
    <emission_b frame_step="1" data_size="1" data_head="13"/>
    <polygon_alpha frame_step="1" data_size="1" data_head="84"/>
  </mat_color_anm>
</mat_color_anm_array>
```

Attributes:

Table 5-21 - The Attributes of the <mat_color_anm> Element

Attribute Name	Description
index	Consecutively numbered. Integer 0 or larger (x1)
material_name	Name of reflective material. Character string (x1)

Contents:Stores each of these elements in this order: <diffuse_r>, <diffuse_g>, <diffuse_b>, <ambient_r>, <ambient_g>, <ambient_b>, <specular_r>, <specular_g>, <specular_b>, <emission_r>, <emission_g>, <emission_b>, and <polygon_alpha>.

5.4.3.5 The <diffuse_r>, <diffuse_g>, and <diffuse_b> Elements

Description: Stores reference information for the diffuse RGB components

Output condition: Outputs only when material color animation has been configured

Output example:

```
<mat_color_anm index="0" material_name="lambert2">
<diffuse_r frame_step="4" data_size="28" data_head="0"/>
<diffuse_g frame_step="4" data_size="28" data_head="28"/>
<diffuse_b frame_step="4" data_size="28" data_head="56"/>
<ambient_r frame_step="1" data_size="1" data_head="13"/>
<ambient_g frame_step="1" data_size="1" data_head="13"/>
<ambient_b frame_step="1" data_size="1" data_head="13"/>
<specular_r frame_step="1" data_size="1" data_head="13"/>
<specular_g frame_step="1" data_size="1" data_head="13"/>
<specular_b frame_step="1" data_size="1" data_head="13"/>
<emission_r frame_step="1" data_size="1" data_head="13"/>
<emission_g frame_step="1" data_size="1" data_head="13"/>
<emission_b frame_step="1" data_size="1" data_head="13"/>
<polygon_alpha frame_step="1" data_size="1" data_head="84"/>
</mat_color_anm>
```

Attributes:**Table 5-22 - The Attributes of the <diffuse_r>, <diffuse_g>, and <diffuse_b> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Size of animation data. Integer 1 or larger (x1) References <mat_color_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <mat_color_data>

Contents: None**5.4.3.6 The <ambient_r>, <ambient_g>, and <ambient_b> Elements****Description:** Stores reference information for the ambient RGB components**Output condition:** Output only when material color animation has been configured

Output example:

```

<mat_color_anm index="0" material_name="lambert2">
<diffuse_r frame_step="4" data_size="28" data_head="0"/>
<diffuse_g frame_step="4" data_size="28" data_head="28"/>
<diffuse_b frame_step="4" data_size="28" data_head="56"/>
<ambient_r frame_step="1" data_size="1" data_head="13"/>
<ambient_g frame_step="1" data_size="1" data_head="13"/>
<ambient_b frame_step="1" data_size="1" data_head="13"/>
<specular_r frame_step="1" data_size="1" data_head="13"/>
<specular_g frame_step="1" data_size="1" data_head="13"/>
<specular_b frame_step="1" data_size="1" data_head="13"/>
<emission_r frame_step="1" data_size="1" data_head="13"/>
<emission_g frame_step="1" data_size="1" data_head="13"/>
<emission_b frame_step="1" data_size="1" data_head="13"/>
<polygon_alpha frame_step="1" data_size="1" data_head="84"/>
</mat_color_anm>

```

Attributes:**Table 5-23 - The Attributes of the <ambient_r>, <ambient_g>, and <ambient_b> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Size of animation data. Integer 1 or larger (x1) References <mat_color_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <mat_color_data>

Contents: None**5.4.3.7 The <specular_r>, <specular_g>, and <specular_b> Elements****Description:** Stores reference information for the specular RGB components**Output condition:** Outputs only when material color animation has been configured

Output example:

```
<mat_color_anm index="0" material_name="lambert2">
<diffuse_r frame_step="4" data_size="28" data_head="0"/>
<diffuse_g frame_step="4" data_size="28" data_head="28"/>
<diffuse_b frame_step="4" data_size="28" data_head="56"/>
<ambient_r frame_step="1" data_size="1" data_head="13"/>
<ambient_g frame_step="1" data_size="1" data_head="13"/>
<ambient_b frame_step="1" data_size="1" data_head="13"/>
<specular_r frame_step="1" data_size="1" data_head="13"/>
<specular_g frame_step="1" data_size="1" data_head="13"/>
<specular_b frame_step="1" data_size="1" data_head="13"/>
<emission_r frame_step="1" data_size="1" data_head="13"/>
<emission_g frame_step="1" data_size="1" data_head="13"/>
<emission_b frame_step="1" data_size="1" data_head="13"/>
<polygon_alpha frame_step="1" data_size="1" data_head="84"/>
</mat_color_anm>
```

Attributes:**Table 5-24 - The Attributes of the <specular_r>, <specular_g>, and <specular_b> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Size of animation data. Integer 1 or larger (x1) References <mat_color_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <mat_color_data>

Contents: None**5.4.3.8 The <emission_r>, <emission_g>, and <emission_b> Elements****Description:** Stores reference information for the emission RGB components**Output condition:** Outputs only when material color animation has been configured

Output example:

```

<mat_color_anm index="0" material_name="lambert2">
<diffuse_r frame_step="4" data_size="28" data_head="0"/>
<diffuse_g frame_step="4" data_size="28" data_head="28"/>
<diffuse_b frame_step="4" data_size="28" data_head="56"/>
<ambient_r frame_step="1" data_size="1" data_head="13"/>
<ambient_g frame_step="1" data_size="1" data_head="13"/>
<ambient_b frame_step="1" data_size="1" data_head="13"/>
<specular_r frame_step="1" data_size="1" data_head="13"/>
<specular_g frame_step="1" data_size="1" data_head="13"/>
<specular_b frame_step="1" data_size="1" data_head="13"/>
<emission_r frame_step="1" data_size="1" data_head="13"/>
<emission_g frame_step="1" data_size="1" data_head="13"/>
<emission_b frame_step="1" data_size="1" data_head="13"/>
<polygon_alpha frame_step="1" data_size="1" data_head="84"/>
</mat_color_anm>

```

Attributes:**Table 5-25 - The Attributes of the <emission_r>, <emission_g>, and <emission_b> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Size of animation data. Integer 1 or larger (x1) References <mat_color_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <mat_color_data>

Contents: None**5.4.3.9 The <polygon_alpha> Element****Description:** Stores the polygon alpha's reference information**Output condition:** Outputs only when material color animation has been configured

Output example:

```

<mat_color_anm index="0" material_name="lambert2">
<diffuse_r frame_step="4" data_size="28" data_head="0"/>
<diffuse_g frame_step="4" data_size="28" data_head="28"/>
<diffuse_b frame_step="4" data_size="28" data_head="56"/>
<ambient_r frame_step="1" data_size="1" data_head="13"/>
<ambient_g frame_step="1" data_size="1" data_head="13"/>
<ambient_b frame_step="1" data_size="1" data_head="13"/>
<specular_r frame_step="1" data_size="1" data_head="13"/>
<specular_g frame_step="1" data_size="1" data_head="13"/>
<specular_b frame_step="1" data_size="1" data_head="13"/>
<emission_r frame_step="1" data_size="1" data_head="13"/>
<emission_g frame_step="1" data_size="1" data_head="13"/>
<emission_b frame_step="1" data_size="1" data_head="13"/>
<polygon_alpha frame_step="1" data_size="1" data_head="84"/>
</mat_color_anm>

```

Attributes:**Table 5-26 - The Attributes of the <polygon_alpha> Element**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Size of animation data. Integer 1 or larger (x1) References <mat_color_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <mat_color_data>

Contents: None**5.4.4 Supplemental Information for ima Files**

- Relation to imd

To play back ima properly, an imd file is required.

imd and ima correspond to the **name** attribute of the imd <material> element and the **material_name** attribute of the ima <mat_color_anm> element.

If the appropriate material names do not exist in the imd file, the ima file data will not be reflected in the imd file.

- Cautions When Playing Using the G3D Library

The G3D library released in the current NITRO-System library does not support the linear interpolation of ima in between frames (interpolation = linear). Even if you set interpolation to linear, play will be the same as if you set frame.

In the future, if the G3D library supports the linear interpolation of in-between frames, setting interpolation to linear will make it possible to play using in-between frames. However, this will increase CPU calculations. To cut down on CPU processing, set interpolation to frame for animation data that does not need Interpolation to be set to linear.

5.5 Texture Pattern Animation Files (itp)

5.5.1 Overview of itp Files

The Texture Pattern Animation file (itp: Intermediate Texture Pattern Animation file) stores data for animation in which the view moves through a series of texture images and palettes.

Texture pattern animation data is output in FV Key format, with the trio of "frame number," "image number," and "palette number" together as a single group.

5.5.2 The Basic Element Structure of itp Files

```
<itp> ... Root
  <head>... Information relating to overall document
    ... (Omitted) ...
  /head>
  <body>... The data itself
    <tex_pattern_info />... Animation settings information
    <tex_pattern_list_data>... Image list & palette list information
      <image_name />... Image list information
      <palette_name />... Palette list information
    /tex_pattern_list_data>
    <tex_pattern_data>... Animation data
      <frame_idx />... Frame data
      <image_idx />... Image data
      <palette_idx />... Palette data
    /tex_pattern_data>
    <tex_pattern_anm_array>... Animation array
      <tex_pattern_anm />... Animation reference information
    /tex_pattern_anm_array>
  /body>
</itp>
```

5.5.3 Explanation of itp Elements and Attributes

5.5.3.1 The <tex_pattern_info> Element

Description: Information relating to overall texture pattern animation is output

Output condition: Always output

Output example:

```
<tex_pattern_info
  frame_size="60"
  tool_start_frame="1"
  tool_end_frame="60"
  compress_material="on" material_size="1 1"
/>
```

Attributes:**Table 5-27 - The Attributes of the <tex_pattern_info> Element**

Attribute Name	Description
frame_size	The number of frames. Integer 1 or larger (x1)
tool_start_frame	The StartFrame specified when outputting file from 3D CG tool. Integer (x1) (For checking the itp output state)
tool_end_frame	The EndFrame specified when outputting file from 3D CG tool. Integer (x1) (For checking the itp output state)
compress_material	The material's compression state. Character string [off / on] off : Output the same number of materials as on tool. on : If materials with exact same settings exist, only output the one that comes first alphabetically. (For checking the itp output state)
material_size	The number of materials when uncompressed and when compressed. Integer 0 or larger (x2) When <code>compress_material=off</code> , the number of uncompressed materials is listed twice (For checking the itp output state)

Contents: None**5.5.3.2 The <tex_pattern_list_data> Element****Description:** Stores the image name and palette name lists used with texture pattern animation**Output condition:** Outputs only when texture pattern animation has been configured**Output example:**

```
<tex_pattern_list_data image_size="3" palette_size="1">
  <image_name index="0" name="p1"/>
  <image_name index="1" name="p2"/>
  <image_name index="2" name="p3"/>
  <palette_name index="0" name="p1_index"/>
</tex_pattern_list_data>
```

Attributes:**Table 5-28 - The Attributes of the <tex_pattern_list_data> Element**

Attribute Name	Description
image_size	The number of images used in this file. Integer 1 or larger (x1)
palette_size	The number of palettes used in this file. Integer 0 or larger (x1)

Contents:Stores *image_size* number of <image_name> elements and *palette_size* number of <palette_name> elements

Image and palette names are ordered in alphanumerical order (a→z and 0→9).

5.5.3.3 The <image_name> Element

Description: Stores image names

Output condition: Outputs only when texture pattern animation has been configured

Output example:

```
<tex_pattern_list_data image_size="3" palette_size="1">
  image_name index="0" name="p1"/>
  <image_name index="1" name="p2"/>
  <image_name index="2" name="p3"/>
  <palette_name index="0" name="p1_index"/>
</tex_pattern_list_data>
```

Attributes:

Table 5-29 - The Attributes of the <image_name> Element

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
name	Image name. Character string (x1)

Contents: None

5.5.3.4 The <palette_name> Element

Description: Stores palette names

Output condition: Outputs only when a texture pattern animation has been configured that uses a texture other than a direct color texture

Output example:

```
<tex_pattern_list_data image_size="3" palette_size="1">
  <image_name index="0" name="p1"/>
  <image_name index="1" name="p2"/>
  <image_name index="2" name="p3"/>
  palette_name index="0" name="p1_index"/>
</tex_pattern_list_data>
```

Attributes:

Table 5-30 - The Attributes of the <palette_name> Element

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
name	Image name. Character string (x1)

Contents:None

5.5.3.5 The <tex_pattern_data> Element

Description: Stores the actual data for playing animation

Output condition: Outputs only when texture pattern animation has been configured

Output example:

```
<tex_pattern_data>
  <frame_idx size="3">
    0 20 40
  </frame_idx>
  <image_idx size="3">
    0 1 2
  </image_idx>
  <palette_idx size="3">
    0 0 0
  </palette_idx>
</tex_pattern_data>
```

Attributes: None**Contents:** Stores one of each of these elements in this order: <frame_idx>, <img_idx>, and <plt_idx>**5.5.3.6 The <frame_idx> Element****Description:** Stores the texture pattern animation's frame numbers

The number of stored data sets and the order correspond to those for <image_idx> and <palette_idx>

Output condition: Output only when texture pattern animation has been configured**Output example:**

```
<tex_pattern_data>
  frame_idx size="3">
    0 20 40
  </frame_idx>
  <image_idx size="3">
    0 1 2
  </image_idx>
  <palette_idx size="3">
    0 0 0
  </palette_idx>
</tex_pattern_data>
```

Attributes:**Table 5-31 - The Attributes of the <frame_idx> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores size number of integers, 0 or larger (frame number)**5.5.3.7 The <image_idx> Element****Description:** Stores the texture pattern animation's image numbers

The number of stored data sets and the order correspond to those for <frame_idx> and <palette_idx>

Output condition: Output only when texture pattern animation has been configured

Output example:

```
<tex_pattern_data>
  <frame_idx size="3">
    0 20 40
  </frame_idx>
  <image_idx size="3">
    0 1 2
  </image_idx>
  <palette_idx size="3">
    0 0 0
  </palette_idx>
</tex_pattern_data>
```

Attributes:**Table 5-32 - The Attributes of the <image_idx> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of integers, 0 or larger (image number)

5.5.3.8 The <palette_idx> Element

Description: Stores the texture pattern animation's palette numbers

The number of stored data sets and the order correspond to those for <frame_idx> and <image_idx>

Output condition: Outputs only when texture pattern animation has been configured

Output example:

```
<tex_pattern_data>
  <frame_idx size="3">
    0 20 40
  </frame_idx>
  <image_idx size="3">
    0 1 2
  </image_idx>
  <palette_idx size="3">
    0 0 0
  </palette_idx>
</tex_pattern_data>
```

Attributes:**Table 5-33 - The Attributes of the <palette_idx> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of integers, -1 or larger (palette number)

(The value -1 is entered if the texture format does not use palettes)

5.5.3.9 The <tex_pattern_anm_array> Element

Description: Stores the texture pattern animation's reference data

Output condition: Outputs only when texture pattern animation has been configured

Output example:

```
<tex_pattern_anm_array size="1">
  <tex_pattern_anm index="0"
    material_name="lambert2" data_size="3" data_head="0"/>
</tex_pattern_anm_array>
```

Attributes:

Table 5-34 - The Attributes of the <tex_pattern_anm_array> Element

Attribute Name	Description
size	The number of <tex_pattern_anm> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <tex_pattern_anm> elements.

5.5.3.10 The <tex_pattern_anm> Element

Description: Stores the texture pattern animation's reference information

Output condition: Outputs only when texture pattern animation has been configured

Output example:

```
<tex_pattern_anm_array size="1">
  tex_pattern_anm index="0"
  material_name="lambert2" data_size="3" data_head="0"/>
</tex_pattern_anm_array>
```

Attributes:

Table 5-35 - The Attributes of the <tex_pattern_anm> Element

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
material_name	The material name in the corresponding imd. Character string (x1)
data_size	Size of animation data. Integer 1 or larger (x1) References the <frame_idx>, <image_idx>, and <palette_idx> elements.
data_head	Location of start of data. Integer 0 or larger (x1) References the <frame_idx>, <image_idx>, and <palette_idx> elements.

Contents: None

5.5.4 Supplemental Information for itp Files

- Relation to imd

To play back itp properly, an imd file is required. imd and itp are matched as follows:

- **Material:** Corresponds to the `name` attribute of the imd `<material>` element and the `material_name` attribute of the itp `<tex_pattern_anm>` element.
- **Image:** Corresponds to the `name` attribute of the imd `<tex_image>` element and the `name` attribute of the itp `<image_name>` element.
- **Palette:** Corresponds to the `name` attribute of the imd `<tex_palette>` element and the `name` attribute of the itp `<palette_name>` element.

When the corresponding material, image, or palette does not exist in the imd file, it is not reflected.

- Image Size and the Number of Palette Colors

When images or palettes are to be substituted in texture pattern animation, you need to equalize (homogenize) the image width and height sizes, the palette format, and the number of colors. If textures with different images sizes or different palette color counts are mixed together, you may not be able to display them correctly.

5.6 Texture SRT Animation Files (ita)

5.6.1 Overview of ita Files

The Texture SRT Animation file (ita: Intermediate Texture SRT Animation file) stores data for animation for which the texture matrix is multiplied by the Scale, Rotate, and/or Translate matrices for texture scaling, rotating, and scrolling.

The texture SRT animation data is output in **Frame format**.

No decision has been made yet whether there will be support for Hermite-interpolated Key format output that many 3D CG tools use.

5.6.2 The Basic Element Structure of ita Files

```

<ita> ... Root
  <head>          ... Information relating to overall document
  (Omitted) ...
  /head>
  <body>          ... The data itself
    <tex_srt_info />... Animation settings information
    <tex_scale_data />... Texture Scale data
    <tex_rotate_data />... Texture Rotate data
    <tex_translate_data /> ... Texture Translate data
    <tex_srt_anm_array>... Animation array
    <tex_srt_anm />... Animation reference information
      <tex_scale_s />... Texture Scale S-direction reference information
      <tex_scale_t />... Texture Scale T-direction reference information
      <tex_rotate />... Texture Rotate reference information
      <tex_translate_s />... Texture Translate S-direction reference information
      <tex_translate_t />... Texture Translate T-direction reference information
    tex_srt_anm />
  /tex_srt_anm_array>
  /body>
</ita>

```

5.6.3 Explanation of ita Elements and Attributes

5.6.3.1 The <tex_srt_info> Element

Description: Information relating to texture SRT animation overall is output

Output condition: Always output

Output example:

```
<tex_srt_info
    frame_size="150"
    tool_start_frame="1"
    tool_end_frame="150"
    interpolation="linear"
    interp_end_to_start="off"
    tex_matrix_mode="maya"
    compress_material="on" material_size="1 1"
    frame_step_mode="auto"
    tolerance_tex_scale="0.01"
    tolerance_tex_rotate="0.5"
    tolerance_tex_translate="0.5"
/>
```

Attributes:

Table 5-36 - The Attributes of the <tex_srt_info> Element

Attribute Name	Description
frame_size	The number of frames. Integer 1 or larger (x1)
tool_start_frame	The StartFrame specified when outputting file from 3D CG tool. Integer (x1) (For checking the ita output state)
tool_end_frame	The EndFrame specified when outputting file from 3D CG tool. Integer (x1) (For checking the ita output state)
interpolation	Specifies the method to use when values from in-between frames are obtained when playing animation. Character string [frame / linear] frame: Discard in-between frames and use integer frames to play. linear: Do not discard in-between frames—use them as they are. When playing, use linear interpolation to obtain in-between frame values. Also refer to "Cautions When Playing Using the G3D Library"
interp_end_to_start Output only when interpolation is linear	Flag indicating whether to perform linear interpolation between final frame and first frame. Character string [off / on] off: Do not change beyond final frame on: Perform linear interpolation

Attribute Name	Description
<code>tex_matrix_mode</code>	The method for computing texture matrices. Character string [maya / si3d] maya : Maya computation method si3d : SOFTIMAGE 3D computation method xsi : SOFTIMAGE XSI computation method 3dsmax : 3ds max computation method For details about these different methods, see Texture Matrix Computation Methods .
<code>compress_material</code>	The material's compression state. Character string [off / on] off : Output the same number of materials as on tool. on : If the materials have the same settings, output only the one that comes first alphabetically (For checking the ita output state)
<code>material_size</code>	The number of materials when uncompressed and when compressed. Integer 0 or larger (x2) When <code>compress_material=off</code> , the number of uncompressed materials is listed twice. (For checking the ita output state)
<code>frame_step_mode</code>	Data culling specification. Character string [1 / 2 / 4 / auto] 1 : Output all frames that are on the 3D CG tool. 2 : Output every other frame (= around 1/2 data size) 4 : Output every fourth frame (= around 1/4 data size) auto : Automatically calculate optimal culling for every animation curve.
<code>tolerance_tex_scale</code>	The tolerable error when culling texture Scale data. Real number 0.0 or larger (x1)
<code>tolerance_tex_rotate</code>	The tolerable error when culling texture Rotate data. Real number 0.0 or larger (x1)
<code>tolerance_tex_translate</code>	The tolerable error when culling texture Translate data. Real number 0.0 or larger (x1)

Contents: None

5.6.3.2 The <tex_scale_data> Element

Description: Stores the texture Scale animation's actual data

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_scale_data size="78">
  1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
  1.000000 1.040039 1.199951 1.360107 1.520020 1.679932 1.840088 2.000000
    ... (Omitted) ...
  2.000000 2.000000 2.000000 2.000000 2.000000 2.000000 2.000000 1.000000
</tex_scale_data>
```

Attributes:**Table 5-37 - The Attributes of the <tex_scale_data> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of real numbers (Scale value)

5.6.3.3 The <tex_rotate_data> Element

Description: Stores the texture Rotate animation's actual data

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_rotate_data size="150">
    0.000000 7.346924 14.693848 22.040771 29.387695 36.734619 44.081543
51.428467 58.775635 66.122559 73.469482 80.816406 88.163330 95.510254
    ... (Omitted) ...
-36.734619 -29.387695 -22.040771 -14.693848 -7.346924 0.000000 0.000000
0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
</tex_rotate_data>
```

Attributes:**Table 5-38 - The Attributes of the <tex_rotate_data> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of real numbers, where -180.0 ≤ r ≤ 180.0 (Rotate value: degrees)

5.6.3.4 The <tex_translate_data> Element

Description: Stores the texture Translate animation's actual data

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_translate_data size="300">
    0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
    20.479980 23.040039 25.600098 28.159912 30.719971 33.280029 35.840088
    ... (Omitted) ...
    64.000000 64.000000 64.000000 64.000000 64.000000 64.000000 64.000000
</tex_translate_data>
```

Attributes:**Table 5-39 - The Attributes of the <tex_translate_data> Element**

Attribute Name	Description
size	The number of data sets stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of real numbers (Translate value)

5.6.3.5 The <tex_srt_anm_array> Element

Description: Stores the texture SRT animation's reference data

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_srt_anm_array size="1">
  <tex_srt_anm index="0" material_name="lambert2">
    <tex_scale_s frame_step="4" data_size="15" data_head="0"/>
    <tex_scale_t frame_step="4" data_size="15" data_head="15"/>
    <tex_rotate frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_s frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_t frame_step="1" data_size="60" data_head="60"/>
  </tex_srt_anm>
</tex_srt_anm_array>
```

Attributes:

Table 5-40 - The Attributes of the <tex_srt_anm_array> Element

Attribute Name	Description
size	The number of <tex_srt_anm> elements stored in this element. Integer 1 or larger (x1)

Contents: Stores *size* number of <tex_srt_anm> elements

5.6.3.6 The <tex_srt_anm> Element

Description: Stores the texture SRT animation's reference information

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_srt_anm_array size="1">
  tex_srt_anm index="0" material_name="lambert2">
    <tex_scale_s frame_step="4" data_size="15" data_head="0"/>
    <tex_scale_t frame_step="4" data_size="15" data_head="15"/>
    <tex_rotate frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_s frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_t frame_step="1" data_size="60" data_head="60"/>
  /tex_srt_anm
</tex_srt_anm_array>
```

Attributes:

Table 5-41 - The Attributes of the <tex_srt_anm> Element

Attribute Name	Description
index	Index number. Integer 0 or larger (x1)
material_name	Reflecting material name. Character string (x1)

Contents: Stores one of each of these elements in this order: <tex_scale_s>, <tex_scale_t>, <tex_rotate>, <tex_translate_s>, and <tex_translate_t>.

5.6.3.7 The <tex_scale_s> and <tex_scale_t> Elements

Description: Stores the texture Scale animation's reference information

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_srt_anm_array size="1">
  <tex_srt_anm index="0" material_name="lambert2">
    tex_scale_s frame_step="4" data_size="15" data_head="0"/>
    <tex_scale_t frame_step="4" data_size="15" data_head="15"/>
    <tex_rotate frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_s frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_t frame_step="1" data_size="60" data_head="60"/>
  </tex_srt_anm>
</tex_srt_anm_array>
```

Attributes:

Table 5-42 - The Attributes of the <tex_scale_s> and <tex_scale_t> Elements

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Amount of animation data. Integer 1 or larger (x1) References <tex_scale_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <tex_scale_data>

Contents: None

5.6.3.8 The <tex_rotate> Element

Description: Stores the texture Rotate animation's reference information

Output condition: Outputs only when texture SRT animation has been configured

Output example:

```
<tex_srt_anm_array size="1">
  <tex_srt_anm index="0" material_name="lambert2">
    <tex_scale_s frame_step="4" data_size="15" data_head="0"/>
    <tex_scale_t frame_step="4" data_size="15" data_head="15"/>
    tex_rotate frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_s frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_t frame_step="1" data_size="60" data_head="60"/>
  </tex_srt_anm>
</tex_srt_anm_array>
```

Attributes:**Table 5-43 - The Attributes of the <tex_rotate> Element**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Amount of animation data. Integer 1 or larger (x1) References <tex_rotate_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <tex_rotate_data>

Contents: None**5.6.3.9 The <tex_translate_s>, <tex_translate_t> Elements****Description:** Stores the texture Translate animation's reference information**Output condition:** Outputs only when texture SRT animation has been configured**Output example:**

```

<tex_srt_anm_array size="1">
  <tex_srt_anm index="0" material_name="lambert2">
    <tex_scale_s frame_step="4" data_size="15" data_head="0"/>
    <tex_scale_t frame_step="4" data_size="15" data_head="15"/>
    <tex_rotate frame_step="1" data_size="60" data_head="0"/>
    tex_translate_s frame_step="1" data_size="60" data_head="0"/>
    <tex_translate_t frame_step="1" data_size="60" data_head="60"/>
  </tex_srt_anm>
</tex_srt_anm_array>

```

Attributes:**Table 5-44 - The Attributes of the <tex_translate_s>, <tex_translate_t> Elements**

Attribute Name	Description
frame_step	Extent of data culling. Integer [1 / 2 / 4] 1: Output all frames 2: Output every other frame (= around 1/2 data size) 4: Output every fourth frame (= around 1/4 data size)
data_size	Amount of animation data. Integer 1 or larger (x1) References <tex_translate_data>
data_head	Location of start of data. Integer 0 or larger (x1) References <tex_translate_data>

Contents: None

5.6.4 Supplemental Information for ita Files

- Relation to imd

To play back ita properly, an imd file is required. imd and ita correspond to the `name` attribute of the imd `<material>` element and the `material_name` attribute of the ita `<tex_srt_anm>` element.

If material of the appropriate names do not exist in the imd file, normal playback will be impossible.

- Methods for calculating texture matrix

The calculation methods for the texture matrix are different for each 3D CG tool. Use the calculation method of the tool specified by `tex_matrix_mode` in the `<tex_srt_info>` element.

- Cautions When Playing Using the G3D Library

The G3D library released from the current NITRO-System library does not support the linear interpolation of in-between frames (interpolation = linear). Even if you set interpolation to linear, play will be the same as if you set frame.

In the future, if the G3D library supports the linear interpolation of ita in between frames, setting interpolation to linear will make it possible to play using in-between frames. However, this will increase CPU calculations. To cut down on CPU processing, set interpolation to frame for animation data that does not need Interpolation to be set to linear.

6 Intermediate Files for Graphic Scenes

6.1 Graphic Scene Data File (isd)

6.1.1 Overview of isd Files

The graphics scene data file (isd: Intermediate Graphics Scene Data file) contains information configured in units of frame when displaying 3D models on NITRO such as camera, lighting, toon table, and fog data.

6.1.2 Specific isd Element Structure

<code><isd></code>	...Root
<code><head></code>	...Information regarding all documents
... (omitted) ...	
<code></head></code>	
<code><body></code>	...Body of the data
<code><camera /></code>	...Camera information
<code><light></code>	...Light information
<code><light0 /></code>	...Light 0 information
<code><light1 /></code>	...Light 1 information
<code><light2 /></code>	...Light 2 information
<code><light3 /></code>	...Light 3 information
<code></light></code>	
<code><shininess table /></code>	...Specular Reflection Table data
<code><toon highlight></code>	...Toon highlight information
<code><toon table /></code>	...Toon table data
<code></toon_highlight></code>	
<code><fog></code>	...Fog information
<code><fog table /></code>	...Fog table data
<code></fog></code>	
<code><alpha test /></code>	...Alpha test information
<code><alpha blending /></code>	...Alpha blending information
<code><y sorting /></code>	...Translucent polygon Y sorting information
<code><edge marking></code>	...Edge marking information
<code><edge color /></code>	...Edge color data
<code></edge_marking></code>	
<code><antialias /></code>	...Antialiasing information
<code><render 1 pixel depth /></code>	...One-pixel polygon display boundary depth information
<code><clear color /></code>	...Clear color information
<code></body></code>	
<code></isd></code>	

6.1.3 isd Elements and Attributes

6.1.3.1 <camera> Element

Description: Stores camera information.

Output Condition: Always outputs 1.

Output Example:

```
<camera
  position="0.000000 0.000000 0.000000"
  lookat="0.000000 0.000000 0.000000"
  twist="0.000000"
  projection="perspective"
  depth_buffer="w"
  scale_w="1.000000"
  near="10.000000"
  far="10000.000000"
  perspective_fovy="30.000000"
  perspective_aspect="1.333333"
  ortho_width="256.000000"
  ortho_height="192.000000"
/>
```

Attributes:**Table 6-1 - <camera> Element Attributes**

Attribute Name	Description
position	Camera position (eyepoint) coordinates. Real number x 3
lookat	Coordinate of the camera focal point . Real number x 3
twist	Degree of tilt in the horizontal direction with respect to the line of sight of the camera. (-180.0 ≤ real number < 180.0) x 1 This indicates how much the camera is rotated with respect to the direction of the camera eyepoint. Positive numbers rotate to the left; negative numbers rotate to the right. Note: If created in the 3D Material Editor, this value will normally be 0.0. This corresponds to the situation where the camera indicated by the camera matrix has an upward vector of (0.0, 1.0, 0.0).
projection	Projection Method. One of [perspective / ortho] perspective: Show as perspective projection. ortho: Show as orthogonal projection.
depth_buffer	Specifies the depth buffering method. [z or w] z: Selects buffering according to the z value. w: Selects buffering according to the w value. This is equivalent to the depth buffering selection flag in the NITRO command SwapBuffers.
scale_w	The scaleW value that is applied to the projection matrix. (Real number greater than 0) x 1.
near	Distance to near clip plane. (Real number ≥ 0.0) x 1
far	Distance to far clip plane. (Real number ≥ 0.0) x 1
perspective_fovy	Vertical (y) visibility angle of perspective projection (0.0 ≤ real number ≤ 180.0) x 1
perspective_aspect	Aspect ratio for perspective projection. (Real number ≥ 0.0) x 1
ortho_width	Projection width for orthogonal projection. (Real number ≥ 0.0) x 1
ortho_height	Projection height for orthogonal projection. (Real number ≥ 0.0) x 1

The following diagram shows the attributes that are stored in the `<camera>` element.

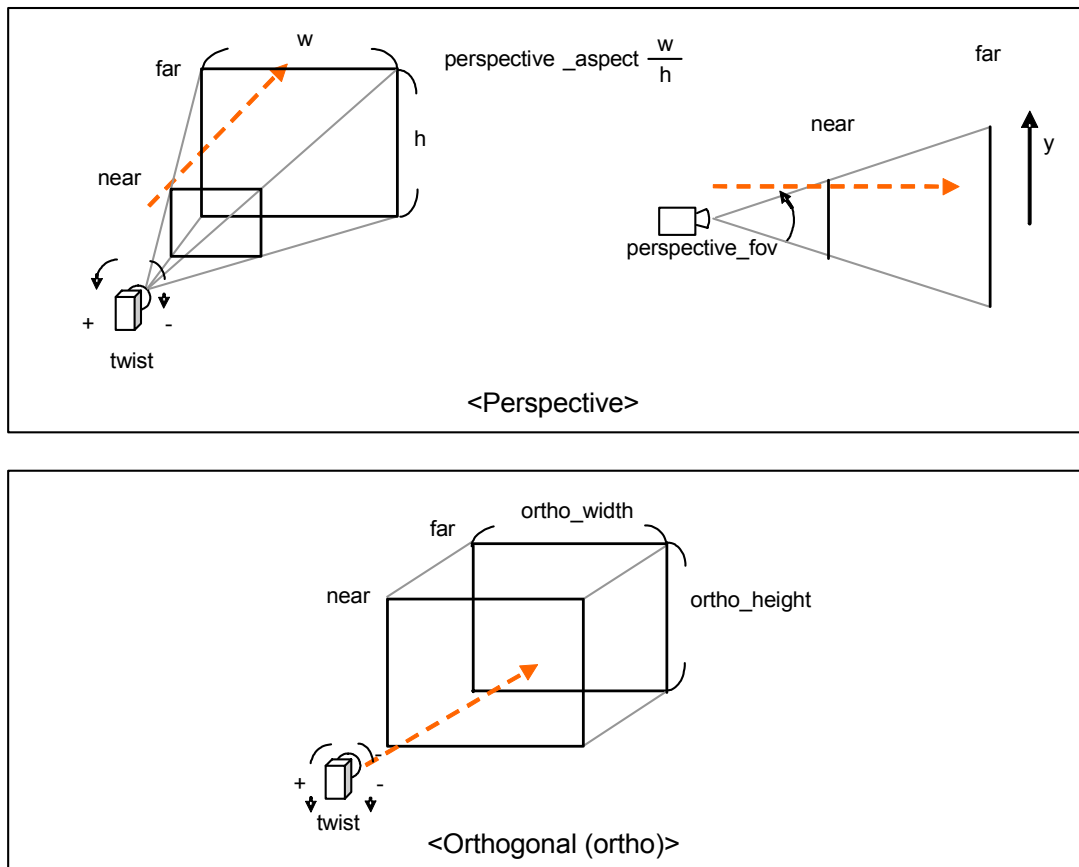


Figure 6-1 - `<camera>` Element Attributes

Contents: None

6.1.3.2 `<light>` Element

Description: Stores light information.

Output Condition: Always outputs 1.

Output Example:

```
<light>
  <light0
    ... (omitted) ...
  />
  <light1
    ... (omitted) ...
  />
  <light2
    ... (omitted) ...
  />
  <light3
    ... (omitted) ...
  />
</light>
```

Attributes: None

Contents: Stores `<light0>`, `<light1>`, `<light2>`, and `<light3>` elements one at a time in this order.

6.1.3.3 <light0>, <light1>, <light2>, and <light3> Elements

Description: Stores information for Light 0, Light 1, Light 2, and Light 3.

Output Condition: Always outputs 1 each.

Output Example:

```
<light>
<light0
direction="-1.000000 0.000000 0.000000"
        color="31 31 31"
/>
<light1
        direction="-1.000000 0.000000 0.000000"
        color="31 0 0"
/>
<light2
        direction="0.000000 -1.000000 0.000000"
        color="0 31 0"
/>
<light3
        direction="0.000000 0.000000 -1.000000"
        color="0 0 31"
/>
</light>
```

Attributes:

Table 6-2 - <light0>, <light1>, <light2>, and <light3> Element Attributes

Attribute Name	Description
direction	Direction vector. $(-1.0 \leq \text{real number} < 1.0) \times 3$ Corresponds to the Direction Vector (X, Y, Z) of the NITRO command <code>LightVector</code> .
color	Color. $(0 \leq \text{integer} \leq 31) \times 3$ Corresponds to the light color (R, G, B) of the NITRO command <code>LightColor</code> .

Contents: None

6.1.3.4 <shininess_table> Element

Description: Stores the Specular Reflection Table data. Corresponds to the Shininess NITRO command.

Output Condition: Always outputs 1.

Output Example:

```
<shininess_table>
    0 2 4 6 8 10 12 14
    ... (omitted) ...
    241 243 245 247 249 251 253 255
</shininess_table>
```

Attributes: None

Contents: Lines up 128 integers in the range of $0 \leq \text{integer} \leq 255$.

Note: Takes the 8-bit unsigned fractional part of the floating-point value, which is used to configure the NITRO Shininess command, and stores it as an integer such that $0 \leq \text{integer} \leq 255$.

6.1.3.5 <toon_highlight> Element**Description:** Stores Toon/Highlight polygon mode information.**Output Condition:** Always outputs 1.**Output Example:**

```

<toon_highlight mode="toon">
  <toon_table>
    (omitted)
  </toon_table>
</toon_highlight>

```

Attributes:**Table 6-3 - <toon_highlight> Element Attributes**

Attribute Name	Description
Mode	<p>Toon/Highlight polygon mode. One of [toon / highlight]</p> <p>toon: Uses toon shading.</p> <p>highlight: Uses highlight shading.</p> <p>Corresponds to the toon/highlight polygon mode of the 3D display control register (DISP3DCNT).</p>

Contents: Stores one <toon_table> element.**6.1.3.6 <toon_table> Element****Description:** Stores toon table data. Corresponds to the toon table register (ToonTable).**Output Condition:** Always outputs 1.**Output Example:**

```

<toon_highlight>
<toon_table>
  0 0 0
  (omitted)
  31 31 31
</toon_table>
</toon_highlight>

```

Attributes: None**Contents:** Lines up 96 (RGBElement x 32) integers in the range of $0 \leq \text{integer} \leq 31$.**6.1.3.7 <fog> Element****Description:** Stores fog information.**Output Condition:** Always outputs 1.

Output Example:

```
<fog
    color="31 31 31"
    alpha="0"
    offset="0"
    shift="0"
    mode="color_alpha"
>
    <fog_table>
        (omitted)
    </fog_table>
</fog>
```

Attributes:**Table 6-4 - <fog> Element Attributes**

Attribute Name	Description
color	RGB component of the fog color. $(0 \leq \text{integer} \leq 31) \times 3$ Corresponds to the fog color register (FogColor) fog color (R, G, B).
alpha	Alpha component of the fog. $(0 \leq \text{integer} \leq 31) \times 1$ Corresponds to the fog color register (FogColor) fog alpha value.
offset	Fog offset. $(0 \leq \text{integer} \leq 32,767) \times 1$ Corresponds to the fog offset register (FogOffset).
shift	Fog shift value. $(0 \leq \text{integer} \leq 10) \times 1$ Corresponds to the fog shift of the 3D display control register (DISP3DCNT).
mode	Fog mode. One of [color_alpha / alpha] color_alpha : Blending with the pixel color value and the alpha-value alpha : Blending with only the pixel alpha-value Corresponds to the fog mode of the 3D display control register (DISP3DCNT).

Contents: Stores one <fog_table> element.

6.1.3.8 <fog_table> Element

Description: Stores fog table data.

Corresponds to the fog density table register (FogTable).

Output Condition: Always outputs 1.

Output Example:

```
<fog_table>
    0 0 0 0 0 0 0 0
    ... (omitted) ...
    0 0 0 0 0 0 0 0
</fog_table>
```

Attributes: None

Contents: Lines up 32 integers in the range of $0 \leq \text{integer} \leq 127$.

6.1.3.9 <alpha_test> Element**Description:** Stores alpha-test information.**Output Condition:** Always outputs 1.**Output Example:**

```
<alpha_test
  enable="off"
  reference="0"
/>
```

Attributes:**Table 6-5 - <alpha_test> Element Attributes**

Attribute Name	Description
enable	alpha-test enable flag. [off / on] Corresponds to the alpha-test enable flag of the 3D display control register (DISP3DCNT).
reference	alpha-test reference value. $(0 \leq \text{integer} \leq 31) \times 1$ Corresponds to the alpha-test reference value of the alpha-test reference value register (AlphaReference).

Contents:None**6.1.3.10 <alpha_blending> Element****Description:** Stores alpha-blending information.**Output Condition:** Always outputs 1.**Output Example:**

```
<alpha_blending enable="off"/>
```

Attributes:**Table 6-6 - <alpha_blending> Element Attributes**

Attribute Name	Description
enable	alpha-blending enable flag. [off / on] Corresponds to the alpha-blending enable flag of the 3D display control register (DISP3DCNT).

Contents: None**6.1.3.11 <y_sorting> Element****Description:** Holds Y-sorting information for translucent polygons.**Output Condition:** Always outputs 1.

Output Example:

```
<y_sorting mode="auto"/>
```

Attributes:**Table 6-7 - <y_sorting> Element Attributes**

Attribute Name	Description
mode	Translucent polygon Y-sorting selection flag. [auto / manual] auto: Auto sort mode manual: Manual sort mode This is equivalent to the translucent polygon Y-sorting selection flag in the NITRO command SwapBuffers.

Contents: None.**6.1.3.12 <edge_marking> Element****Description:** Stores edge marking information.**Output Condition:** Always outputs 1.**Output Example:**

```
<edge_marking enable="off"/>
  <edge_color>
    31 31 31
    ... (omitted) ...
    0 0 0
  </edge_color>
</edge_marking>
```

Attributes:**Table 6-8 - <edge_marking> Element Attributes**

Attribute Name	Description
Enable	Edge marking enable flag. [off / on] Corresponds to the edge marking enable flag of the 3D display control register (DISP3DCNT).

Contents: Store one <edge_color> element.**6.1.3.13 <edge_color> Element****Description:** Store edge color data. Corresponds to the edge color register (EdgeColor)**Output Condition:** Always outputs 1.

Output Example:

```
<edge_color>
  31 31 31
  ... (omitted) ...
  31 0 0
</edge_color>
```

Attributes: None**Contents:** Lines up 24 integers (RGBElement x 8) in the range of $0 \leq \text{integer} \leq 31$.**6.1.3.14 <antialias> Element****Description:** Stores antialiasing information.**Output Condition:** Always outputs 1.**Output Example:**

```
<antialias enable="off"/>
```

Attributes:**Table 6-9 - <antialias> Element Attributes**

Attribute Name	Description
enable	Antialiasing enable flag. [off / on] Corresponds to the antialiasing enable flag of the 3D display control register (DISP3DCNT).

Contents: None**6.1.3.15 <render_1_pixel_depth> Element****Description:** Holds the one-pixel (dot) polygon display boundary depth value.**Output Condition:** Always outputs 1.**Output Example:**

```
<render_1_pixel_depth w="2048.000000"/>
```

Attributes:**Table 6-10 - <render_1_pixel_depth> Element Attributes**

Attribute Name	Description
w	One-pixel (dot) polygon display boundary depth value. (Real number greater than or equal to 0) x 1. This is equivalent to the one-dot polygon display boundary depth value register (Disp1DotDepth).

Contents: None**6.1.3.16 <clear_color> Element****Description:** Stores clear color information.

Output Condition: Always outputs 1.

Output Example:

```
<clear_color  
  color="31 31 31"  
  alpha="0"  
  polygon_id="0"  
  fog_enable="off"  
  depth="0"  
>
```

Attributes:

Table 6-11 - <clear_color> Element Attributes

Attribute Name	Description
color	Clear color value. $(0 \leq \text{integer} \leq 31) \times 3$ Corresponds to clear color (R, G, B) of the clear color attribute register (ClearColorAttr).
alpha	Alpha initial value. $(0 \leq \text{integer} \leq 31) \times 1$ Corresponds to alpha initial value of the clear color attribute register (ClearColorAttr).
polygon_id	Polygon ID initial value. $(0 \leq \text{integer} \leq 63) \times 1$ Corresponds to the Polygon ID initial value of the clear color attribute register
fog_enable	Fog enable flag. Either off or on. Corresponds to fog enable flag of the clear color attribute register
depth	Clear depth value. $(0 \leq \text{integer} \leq 32,767) \times 1$ Corresponds to the clear depth register (ClearDepth).

Contents: None

6.1.4 isd Supplement

- Outputting Camera and Light Animation Data

The values of the elements that are contained in the camera and lighting graphics scenes in the various 3D CG tools and the output of the intermediate files for related animation data have yet to be determined.

7 Optimizing Data

This section provides supplemental explanations about data optimization when outputting data from the 3D CG tool to Intermediate files.

7.1 Culling Nodes

As the number of nodes increases, so does the burden involved in performing matrix calculations for the display of 3D models. When you use the NITRO Intermediate File Plug-in to output data from the 3D CG tool to an imd file, you can lighten this burden and conserve on memory space by optimizing the data by culling nodes that are not needed to display the 3D model.

The table shows the kinds of node culling you can specify with the NITRO Intermediate File Plug-in.

For details, see the individual NITRO Intermediate File Plug-in manuals for each 3D CG tool.

Table 7-1 - Types of Node Culling

Mode	Description
None	Nodes are not culled. The data are output with the same hierarchical structure created on the 3D CG tool.
Cull Useless Node	Data are output after culling nodes that are not needed for model display. Effective for character models that use skinning (envelopes).
Merge Useless Node	In addition to the "Cull Useless Node" process, nodes that are not needed for drawing are merged together. This additional process results in even fewer nodes. However, there is a restriction: for nodes with children, you must configure a non-uniform Scale (i.e., one where the Scale x, y, and z values are not the same). If you do not comply with this restriction, the model may not be displayed properly. Even though this mode comes with a restriction, it is the best way to minimize the number of nodes for character models.
Unite	Nodes are combined into a single node and all polygons are output in global coordinates. However, polygons that originally belonged to separate nodes (imd <polygon> elements) are output separately even if the material is the same. This mode is effective when you want to cull the topographic data portion by portion.
Unite and Combine Polygon	In addition to the Unite process, polygons that display in the same material are combined. However, even if the material is the same, polygons will not be combined if either the presence or absence of vertex color changes, or the priority changes. You can expect faster display speed than Unite, but you lose the capacity for detailed clipping. This mode is effective for topographic data when you want to reduce the data size and the process load as much as possible.

7.2 Material Compression

Materials that have different names but the same configuration can be combined together to reduce the size of the imd file and boost processing efficiency on NITRO. If materials that meet these conditions exist, only the file that comes first alphabetically is output. Note that compression is not possible for materials configured for material color animation, texture pattern animation, and texture SRT animation because separate materials need to exist for these.

7.3 Making Polygon Strips

When outputting from the 3D CG tool to the intermediate format, if you set the `use_primitive_strip` function on the output dialog to ON, the polygon stripping process (using triangular polygon strips and quadrilateral polygon strips) in the plug-in automatically starts and is optimized to draw polygons with as few NITRO geometry commands as possible.

When polygons are not in strips (when the above function is set to OFF), all polygons are displayed as either triangular or quadrilateral polygons.

Regardless of usage or non-usage of polygon strips there is no change in the display of the model, but the strip version has fewer processing vertices, which reduces the data size.

8 Reference Information

This section provides information for programmers who will be using Intermediate files to construct their own display routines.

8.1 Matrix Computations

8.1.1 Node Matrix Computation Methods

Basic Matrix Calculation Method

The matrix computation method for node C of a model with the hierarchical structure A -> (parent) B -> (child) C (grandchild) is as follows:

$$[Sc] * [Rc] * [Tc] * [Sb] * [Rb] * [Tb] * [Sa] * [Ra] * [Ta]$$

Each 3D CG tool may use different matrix computations apart from this method.

Maya's Segment Scale Compensation

With Maya, you can perform a particular kind of matrix calculation which uses the Segment Scale Compensate attribute. For details about this attribute, refer to the NITRO Intermediate File Plug-in for Maya or the Maya manual.

SOFTIMAGE|3D's Classic Scaling

SOFTIMAGE|3D has a switch called Classic Scaling that is used for scenes. By using the switch, models can be rendered using SOFTIMAGE's proprietary matrix calculations instead of the usual matrix calculation. Refer to the NITRO Intermediate File Plug-in for SOFTIMAGE|3D manual for details for displaying models in the same manner as Softimage proprietary matrix calculations.

SOFTIMAGE|XSI's Hierarchical Scaling

SOFTIMAGE|XSI includes a switch named Hierarchical Scaling. When this is ON, as with SOFTIMAGE|3D's Classic Scaling, models can be rendered using Softimage's proprietary matrix calculations.

With SOFTIMAGE|XSI, it is also possible to change Hierarchical Scaling settings at each node just as with SOFTIMAGE|3D's Classic Scaling. We are treating this the same as an overall model setting item.

Refer to the NITRO Intermediate File Plug-in for SOFTIMAGE|XSI manual for details about displaying models in the same manner as Softimage proprietary matrix calculations (Hierarchical Scaling ON).

8.1.2 Texture Matrix Computation Methods

The method for computing texture matrices differs among 3D CG tools. For details about the texture matrix calculation method that displays the same in each 3D CG tool, see the NITRO Intermediate File Plug-in Manual or the manuals supplemental to each 3D CG tool.

