

A Description of the NITRO 4×4 Texel Compressed Texture

Version 1.0.0

Published by Nintendo Co., Ltd

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

Contents

1	Introduction	5
2	Format of 4x4 texel compressed texture	6
2.1	Organization of 4x4 texel compressed texture	6
2.1.1	Texel data	7
2.1.2	Palette index data	7
2.1.3	Texture color data	9
2.2	Texture size and amount of memory consumed	10
2.3	Memory configuration of 4x4 texel compressed texture format	10
2.3.1	Basic memory configuration	10
2.3.2	Memory configuration of texel data and palette index data	11
2.3.3	Memory configuration of texture palette	12
2.3.4	Precautions for memory configuration of texture palette	12
2.3.5	Precautions when compressing a texture palette	13
2.3.6	Efficiently compressing a texture palette	14
3	Advantages and disadvantages of 4x4 texel compressed texture	15
3.1	Advantages of 4x4 texel compressed texture	15
3.1.1	Reducing data size	15
3.1.2	Image quality	15
3.1.3	Development speed	19
3.2	Disadvantages of 4x4 texel compressed texture	19
3.2.1	Transparency limitations	19
3.2.2	Memory consumption of texture palette	19
4	Comparison with S3TC compressed texture	20
4.1.1	S3TC compressed texture	20
4.1.2	Differences from 4x4 texel compressed texture	20
4.1.3	Image quality	20
5	Creating a 4x4 texel compressed texture	21
5.1	Tools that can create a 4x4 texel compressed texture	21
5.1.1	ntexconv	21
5.2	Creating a converter	21
5.2.1	TXLib	21
6	Conclusion	22

Tables

Table 2-1	Memory consumption comparison table for each texture format	10
Table 2-2	Example of color sequence to efficiently compress a texture palette	14

Figures

Figure 2-1	Outline of 4x4 texel compressed texture organization	6
Figure 2-2	Texel data format	7
Figure 2-3	Palette index data format	8
Figure 2-4	Texture color data format	9
Figure 2-5	Memory configuration of 4x4 texel compressed texture	11
Figure 2-6	Memory configuration of texture palette	12
Figure 2-7	Example of texture palette compression failure when mixing with / without linear interpolation	13
Figure 3-1	Comparison of bitmap images (32x32 texel)	16
Figure 3-2	Comparison of gradation (128x128 texel)	16
Figure 3-3	Comparison of CG image (256x256 texel)	17
Figure 3-4	Comparison of images of nature (256x256 texel)	18
Figure 4-1	Comparison of S3TC and bitmap images of 4x4 texel compressed texture (32x32 texel)	20

Revision History

Version	Revision Date	Revision Contents
1.0.0	02-03-2004	Release

1 Introduction

This document describes the 4x4 texel compressed texture, one of the NITRO texture formats. This document is based on the *NITRO Programming Manual*, Version 0.10.

2 The Format of the 4x4 Texel Compressed Texture

2.1 Organization of 4x4 Texel Compressed Texture

As the name implies, the 4x4 texel compressed texture divides the texture map into 4x4 texel blocks. Each texel block consists of three types of data: texel data, palette index data, and texture color data.

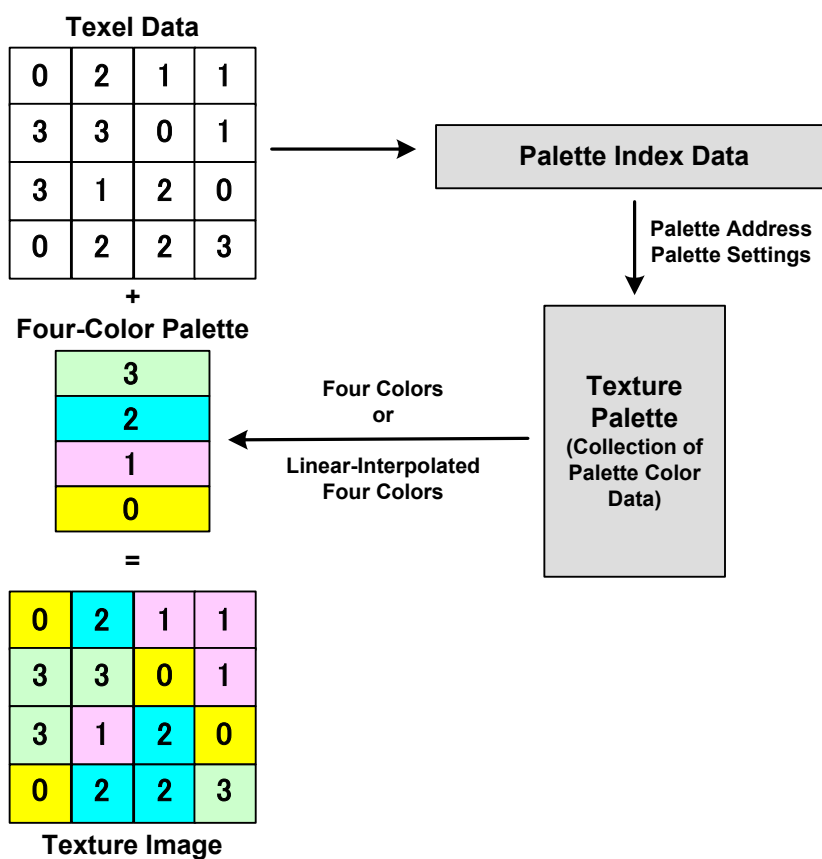


Figure 2-1 An Outline of the 4x4 Texel Compressed Texture Organization

2.1.1 Texel Data

Texel data is color index data held by each texel. There are 2 bits per texel, and each has a color index from 0 ~ 3. This data indicates which color data to use from the four colors generated by the palette index data to be referenced. The texel data must have $4 \times 4 = 16$ texel units (32-bit units).

31	24			23	16			15	8			7	0		
T33	T32	T31	T30	T23	T22	T21	T20	T13	T12	T11	T10	T03	T02	T01	T00
Data of 4x4 texel portion (2-bit/texel)															

●T33~T00: Texel data

Specifies color No. (0 ~ 3).

Display Texture

T00	T01	T02	T03				
T10	T11	T12	T13				
T20	T21	T22	T23				
T30	T31	T32	T33				

Figure 2-2 Texel Data Format

The texel block that will reference a palette index is determined by each respective data memory configuration. Refer to “2.3.2 Memory Configuration of Texel Data and Palette Index Data” for details.

2.1.2 Palette Index Data

Palette index data is referenced from the texel data. The palette index data has texture palette address data that determines which texture color data in the texture palette will be referenced. It also has palette setting data that specifies how to use that texture color data. There must be one piece of palette index data for each texel block (palette index data cannot be shared by multiple texel blocks). This is because texel data from one texel block and one piece of palette index data always form a set.

15	14	13	8			7	0		
A	PTY								
Palette settings		Palette address							

Palette settings

●A: 3-color / 4-color setting flag

0	3-color + transparent color mode (Makes color 3 transparent)
1	4-color mode

●PTY: Palette type selection flag

0	4-color palette (4 palettes per each 4x4 texel)
1	Linear interpolation 4-color palette (2 palettes per each 4x4 texel)

●Palette address

Specifies the address of the texture palette slot where color data is stored using 2-color units (4 byte units). In short, this address regards the color located at the address signified by (value set by `TexPlttBase` command X 0x10) + (palette address set value X 4) in the texture palette slot as color 0.

The color value of a texel is calculated for each RGB as shown in the table below.

	PTY=0	PTY=1
A=0	Color 0: No process Color 1: No process Color 2: No process Color 3: Transparent	Color 0: No process Color 1: No process Color 2: (Color 0+Color 1)/2 Color 3: Transparent
A=1	Color 0: No process Color 1: No process Color 2: No process Color 3: No process	Color 0: No process Color 1: No process Color 2: (5×Color 0+3×Color 1)/8 Color 3: (3×Color 0+5×Color 1)/8

Figure 2-3 Palette Index Data Format

Palette index data uses palette settings to reference 2-color or 4-color texture color data from the address of a specified texture palette.

When a 4-color palette has been selected for the palette type setting, texture color data of a continuous 4-color portion is referenced from the address of the specified texture palette and assigned to 0 ~ 3 of each texel data. When the 3-color + transparent color mode is set at this time by the 3-color/4-color setting flag, Color 3 will be transparent regardless of the color value of Color 3.

When a linear interpolated 4-color palette has been selected for the palette type setting, the texture color data only of the continuous 2-color portion is referenced from the address of the specified texture palette and four linear interpolated colors are created using the formulas in the table above. As described above, when the 3- color + transparent color mode is set by the 3-color/4-color setting flag, color 3 will be transparent regardless of the color value of Color 3.

2.1.3 Texture Color Data

Texture color data is shared with data used by other NITRO formats and has color data that is used in textures. A single color is comprised of 5-bit components of red, green, and blue (total of 32,768 colors).

15	14	10	9	8	7	5	4	0
	TEX_COLOR_BLUE			TEX_COLOR_GREEN			TEX_COLOR_RED	
	Color							

Figure 2-4 Texture Color Data Format

Texture color data is stored in a texture palette. Because the texture palette is referenced from the palette index data, multiple palette index data can reference identical texture color data. Referencing identical texture color data from multiple palette index data reduces the amount of texture color data used. In other words, you can compress a texture palette.

Texture color data is stored in a specified VRAM as a texture palette. Refer to “2.3.3 Memory Configuration of Texture Palette” for details.

2.2 Texture Size and Amount of Memory Consumed

Texture sizes that can be used by NITRO are 8x8 texel ~ 1024x1024 texel. The following table compares memory consumption by each type of texture format that can be used by NITRO.

Table 2-1 Memory Consumption Comparison Table for Each Texture Format

	8x8	16x16	32x32	64x64	128x128	256x256	512x256	512x512	1024x512	1024x1024
4-color texture	16	64	256	1K	4K	16K	32K	64K	128K	256K
16-color texture	32	128	512	2K	8K	32K	64K	128K	256K	512K
256-color texture	64	256	1K	4K	16K	64K	128K	256K	512K	×
Translucent texture (A5I3, A3I5)	64	256	1K	4K	16K	64K	128K	256K	512K	×
Direct color texture	128	512	2K	8K	32K	128K	256K	512K	×	×
4x4 texel compression	Texel data	16	64	256	1K	4K	16K	32K	64K	128K
	Palette index	8	32	128	512	2K	8K	16K	32K	64K
	Palette (Interpolation)	16	64	256	1K	4K	16K	32K	64K	(96K)
	Palette (No interpolation)	32	128	512	2K	8K	32K	64K	(96K)	(96K)
	Total (Interpolation)	40	160	640	2.5K	10K	40K	80K	160K	(288K)
	Total (No interpolation)	56	224	896	3.5K	14K	56K	112K	(192K)	(480K)

Units: Bytes

×: Not in specifications

Parenthesis: Indicates value when texture palette is compressed because the maximum value will exceed RAM capacity.

4-, 16-, and 256-color textures and translucent textures are not considered in the size of the palette.

The amount of memory consumed by a 4x4 texel compressed texture palette is when there is no compression.

The amount of memory a texture palette uses will change for a 4x4 texel compressed texture depending on whether the texture color data undergoes linear interpolation (creates 4 colors from 2 colors) or not (uses 4 colors). For either case, the amount of memory used can be reduced more than shown in the table above by efficiently compressing the texture palette.

If a certain texture size exceeds the limit of VRAM that can be handled as a texture palette, the size of the texture palette must be compressed to 96 kilobytes. The range of texture sizes for palettes that do not need to be compressed can be seen in the table above. When there is no linear interpolation, the size is 512x256 or less; when there is linear interpolation, the size is 512x512 or less.

2.3 Memory Configuration of 4x4 Texel Compressed Texture Format

Be sure to pay attention to the memory configuration when using a 4x4 texel compressed texture. For 4x4 texel compressed textures, texel data for a 4x4 texel (one texel block) requires one piece of palette index data. The correspondence between them is determined by the memory configuration.

2.3.1 Basic Memory Configuration

Texel data must be located in texture image slots 0 and 2. Palette index data must be located in texture

image slot 1.

2.3.2 Memory Configuration of Texel Data and Palette Index Data

Place the texture palette index data that corresponds to 4x4 texel data located at address TIAa of texture image slot 0 in the address of texture image slot 1 ($TIAa/2$). Also place texture palette index data that corresponds to 4x4 texel data located at address TIAb of texture image slot 2 in the address of texture image slot 1 ($0x10000 + (TIAb/2)$). (Refer to the figure below.)

For each texel color, the color of the address (texture color palette slot) specified by texture palette index data is designated as number 0, and the color of the color number specified by the texel data is used.

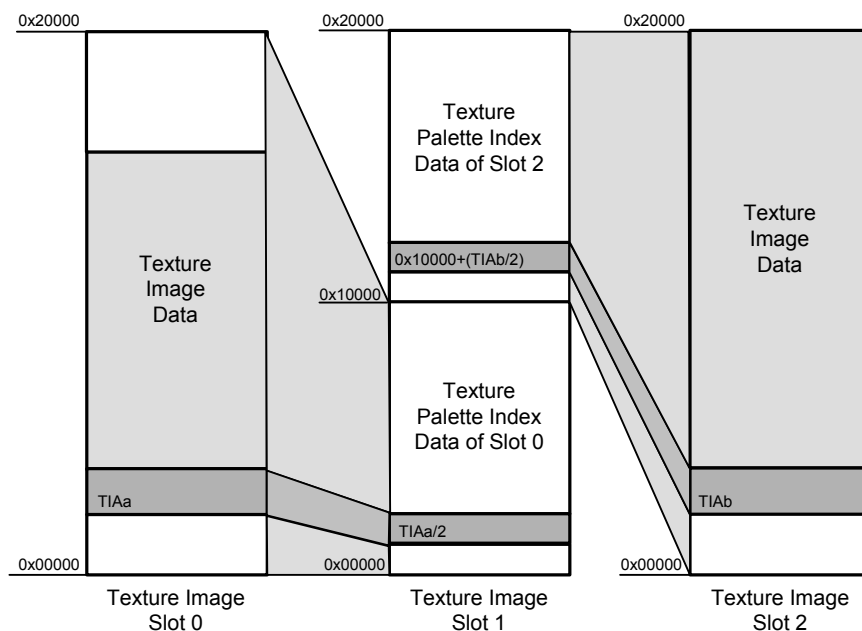


Figure 2-5 Memory Configuration of the 4x4 Texel Compressed Texture

2.3.3 Memory Configuration of the Texture Palette

Texture color data is stored in the texture palette slot.

The texture color palette determines the reference method according to the texture format (4-color / 256-color / 4x4 texel compressed / translucent).

Basically, the color of the color number specified by the texel data is referenced from the palette specified by the texture palette base. However, for a 4x4 texel compressed texture, the palette is specified by setting the palette address in addition to the texture palette base. See the figure below.

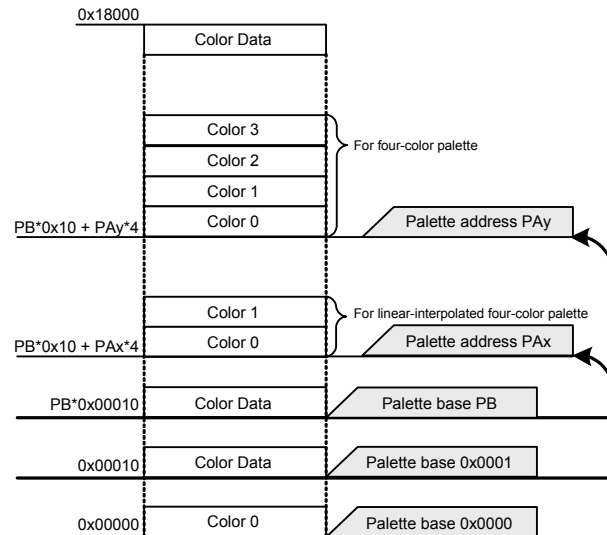


Figure 2-6 Memory Configuration of the Texture Palette

For a 4x4 texel compressed texture, the palette base and palette address specify the address of the palette. For a 4-color palette, color 0 - color 3 are referenced although for a linear interpolated 4 color palette, only color 0 and color 1 are referenced. The remaining 2 colors are calculated from color 0 and color 1 and applied to the texel.

Further, when “3-color + transparent color mode” is used for a 4-color palette, color 3 is not referenced and becomes transparent.

2.3.4 Precautions for Memory Configuration of the Texture Palette

Texture palettes stored in RAM can be shared by all textures. Not only can a 4x4 texel compressed texture be shared but also palettes of index textures such as 16-color textures and 256-color textures. Because of this, you must be careful of the boundary of the texture palette used for each texture when placing color data in a texture palette slot.

2.3.5 Precautions when Compressing a Texture Palette

You must be careful of the boundary of 2-color units when compressing a texture palette that is only a 4x4 texel compressed texture and has both linear interpolated and non-interpolated data. The reason for this is because the address that indicates the texture palette is in 2-color units per address. When there is linear interpolated data, the texture color data only uses 2 colors and only requires 1 address portion of the palette. When there is no linear interpolated data, the texture color data uses 4 colors and requires 2 address portions of the palette (since a continuous 2 address portion is used, one address is specified).

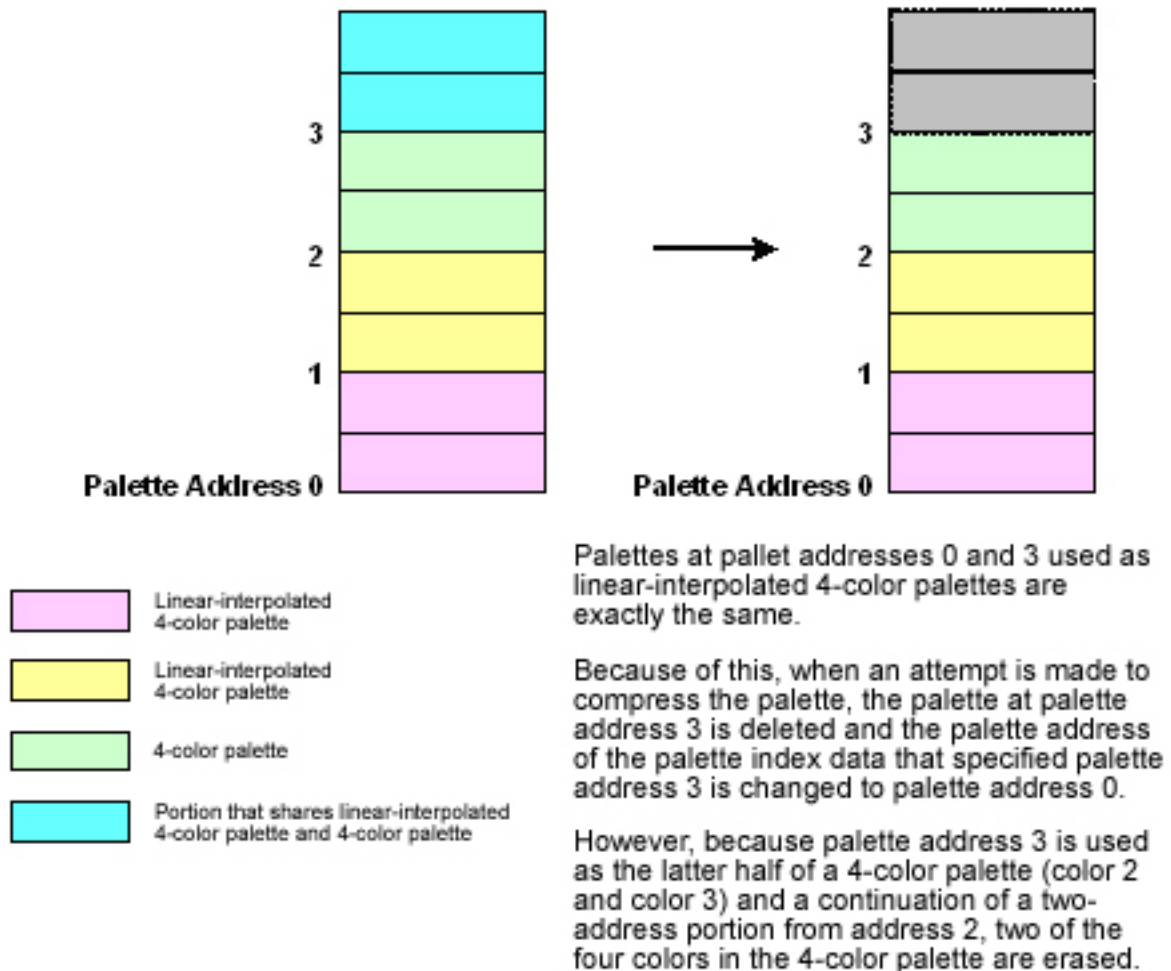


Figure 2-7 Example of Texture Palette Compression Failure when Mixing with / without Linear Interpolation

2.3.6 Efficiently Compressing a Texture Palette

The color order for each palette is important in order to efficiently compress a texture palette. For example, if the color order for the palette is organized as shown below, it will become easy to find an identical (or similar) palette, thereby making it possible to efficiently compress the palette.

Table 2-2 Example of a Color Sequence to Efficiently Compress a Texture Palette

Color 0	Color with highest degree of brightness
Color 1	Color with lowest degree of brightness
(Color 2)	(Color with 2 nd highest degree of brightness)
(Color 3)	(Color with 3 rd highest degree of brightness)

Even when there is a mix of both linear interpolated and non-interpolated data, color 0 and color 1 will be easier to match (or to find an approximation) by placing the colors in the order shown above. Texture palettes can be compressed if the same palette is used when it match (or approximates) another palette.

3 Advantages and Disadvantages of the 4x4 Texel Compressed Texture

3.1 Advantages of the 4x4 Texel Compressed Texture ---

3.1.1 Reducing Data Size ---

Palette addresses referenced from palette index data are shared by multiple pieces of palette index data. In other words, you can compress texture palettes by using the same texture color data with multiple pieces of palette index data. Texel blocks that use the same texture data can also be used as four individual colors (two colors are shared) according to the palette settings for texture palettes. The greatest advantage of a 4x4 texel compressed texture is the fact that the texture data size can be reduced by efficiently compressing texture palettes in this manner.

3.1.2 Image Quality ---

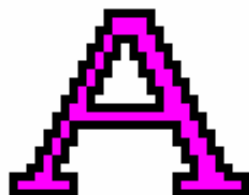
Since interpolation allows the two colors of a texture palette to be used as fours, high quality is maintained while keeping the data size small.

In addition, when interpolation is not used a 4-color palette is for each texel block. This allows high quality reproductions to be maintained for a wide range of image types including images with sharp color variations, such as bitmap images, images with smooth color variation gradations, and natural images.

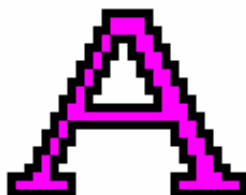
Furthermore, since a combination of interpolation and non-interpolation can be used in one image, data sizes can be efficiently reduced while maintaining high image quality.

The following tables show examples of comparisons of 4x4 texel compressed textures and other NITRO texture formats. The examples shown below are only examples. Please be aware that image quality and texture palette data size may vary for 4x4 texel compressed textures due to the compression algorithm.

Direct color texture



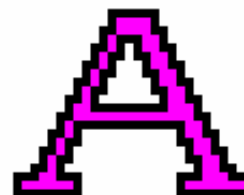
16 color texture



256 color texture



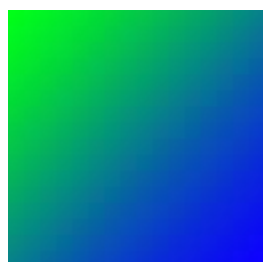
4x4 texel compressed texture



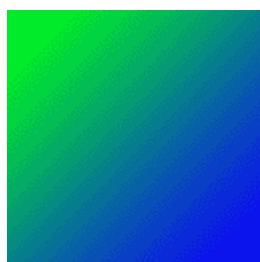
Texel data	4KB	Texel data	512 bytes	Texel data	1 kilobyte	Texel data	256 bytes
		Palette	6 bytes	Palette	6 bytes	Palette index	128 bytes
						Palette	6 bytes
Total	4KB	Total	518 bytes	Total	App. 1 kilobyte	Total	390 bytes

Figure 3-1 A Comparison of Bitmap Images (32x32 Texel)

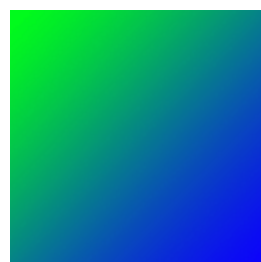
Direct color texture



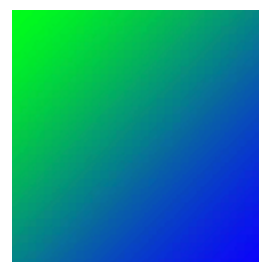
16-color texture



256-color texture


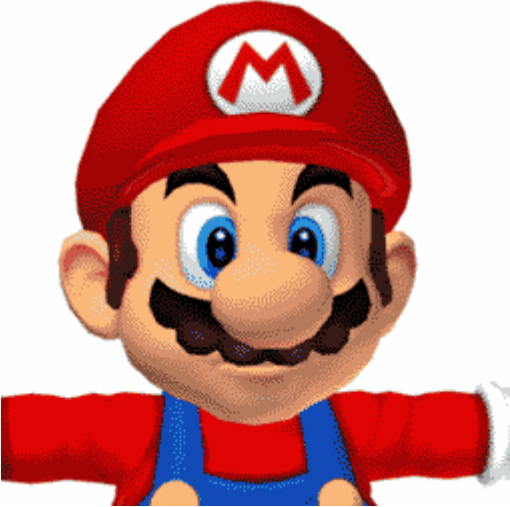


4x4 texel compressed texture



Texel data	32 kilobytes	Texel data	8 kilobytes	Texel data	16 kilobytes	Texel data	4 kilobytes
		Palette	26 bytes	Palette	506 bytes	Palette index	2 kilobytes
						Palette	948 bytes
Total	32 kilobytes	Total	App. 8 kilobytes	Total	App.16.5 kilobytes	Total	App. 7 kilobytes

Figure 3-2 A Comparison of Gradation (128x128 Texel)

Direct color texture		16-color texture	
			
15126 colors used		16 colors used	
Texel data	128 kilobytes	Texel data	32 kilobytes
		Palette	32 bytes
Total	128 kilobytes	Total	App. 32 kilobytes



256-color texture		4x4 texel compressed texture	
			
256 colors used		2265 colors used	
Texel data	64 kilobytes	Texel data	16 kilobytes
Palette	512 bytes	Palette index	8 kilobytes
		Palette	9.6 kilobytes
Total	64.5 kilobytes	Total	33.6 kilobytes

Figure 3-3 A Comparison of CG Images (256x256 Texel)



Figure 3-4 A Comparison of Natural Images (256x256 Texel)

3.1.3 Expansion Speed

Because 4x4 texel compressed textures are expanded in real time on hardware, expansion can be accomplished quickly without any penalties when compared to other texture formats.

3.2 Disadvantages of the 4x4 Texel Compressed Texture

3.2.1 Transparency Limitations

A 4x4 texel compressed texture can have only opaque texels or fully transparent texels. If you need a multi-level degree of transparency for a texture in NITRO, you must use the A513 or A315 translucent texture format.

3.2.2 Texture Palette Memory Consumption

Even if you use a 4x4 texel compressed texture to reduce the entire data size of a texture, you will find that in almost all cases, the amount of memory used by the texture palette is more than other NITRO texture formats. Because of this, you must pay attention to the amount of memory used by the texture palette. Since a texture palette can be shared with other texture formats, a 4x4 texel compressed texture should become even easier to use if shared efficiently.

4 A Comparison with the S3TC Compressed Texture

4.1.1 The S3TC Compressed Texture

A compressed texture format similar to the 4x4 texel compressed texture format is S3TC (S3's Texture Compression), a texture compression technology developed by the S3 company. Like the 4x4 texel compressed texture, S3TC divides texture image files into blocks of 4x4 texel units and then reduces the colors.

4.1.2 Differences from the 4x4 Texel Compressed Texture

The major differences between a 4x4 texel compressed texture and S3TC are as follows.

- In the S3TC format, two representative colors are determined for each texel block and the 4 colors of the palette are created by using linear interpolation on those two colors. The 4x4 texel compressed texture supports linear interpolation like S3TC does but also allows four colors to be used in the palette without using linear interpolation.
- The S3TC format has one palette for each texel block. However, the 4x4 texel compressed texture not only indexes colors but also indexes palettes. This makes it possible to share palettes among texel blocks.

From the descriptions above, the number of S3TC palettes is determined by the image size. However, the number of palettes for 4x4 texel compressed texture can vary if the image changes even if the image size is the same. Even with the same image, the number of palettes will vary according to the compression algorithm.

4.1.3 Image quality

The S3TC compressed texture has noticeable image degradation when there are sharp color variations such as in bitmap images. 4x4 texel compressed texture does not experience this degradation.

S3TC compressed texture



4x4 texel compressed texture



Figure 4-1 A Comparison of S3TC and Bitmap Images of 4x4 Texel Compressed Textures (32x32 Texel)

5 Creating a 4x4 Texel Compressed Texture

5.1 Tools that can Create a 4x4 Texel Compressed Texture ---

This section introduces tools that can create 4x4 texel compressed texture.

5.1.1 ntexconv ---

This is a sample tool created with TXLib (described later) that can convert BMP or TGA image files to NTF. NTF stands for NITRO Texture File and refers to files appropriate for use in NITRO programs. Refer to the ntexconv manual for details.

5.2 Creating a Converter ---

The following library is provided for the purpose of creating texture data for NITRO.

5.2.1 TXLib ---

TXLib is a library that converts BMP or TGA image files, to NTF. Refer to the TXLib manual for details.

6 Conclusion

The 4x4 texel compressed texture is a texture format that makes it possible to obtain the smallest data size while maintaining high image quality for all types of images. If the texture format can be efficiently compressed, the texture format is easier to use. The 4x4 texel compressed texture gives NITRO better 3D graphics quality.

© 2003, 2004 Nintendo

The contents of this document cannot be duplicated, copied, reprinted, transferred, distributed or loaned in whole or in part without the prior approval of Nintendo.