Alpha Compression with Variable Data Formats

Yifei Jiang[†1,2] Mindan Gui[3] Shuai Chen[1,2] Weiwu Hu[1]

[1]Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China
[2]Graduate University of Chinese Academy of Sciences, Beijing 100049, China
[3]Department of Electronic Engineering, Wuxi Nanyang Vocational College, Wuxi 214081, China

Abstract

Based on the insight into the essential differences between transparency and color, we present a novel alpha compression scheme, called EAC (Enhanced Alpha Compression), as an improvement to that of DXT5. EAC defines new compression data formats, and employs clustering algorithms combined with linear interpolation method used in DXT5 to compress 16 input alpha values to 64 bits. The compressed block has three possible formats, corresponding to two process modes in decompression. Results show that EAC improves the average PSNR with about 0.2 dB compared to alpha compression of DXT5 over a set of test images. Moreover, the average cell area of hardware implementation is reduced by about 25%, and the power consumption is lowered by 21.66%–26.75%.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Texture

1. Introduction

Memory bandwidth is often the bottleneck for graphics systems [AMHH08], as the yearly performance growth rate of computing capability is much larger than that of bandwidth and latency for DRAM [Owe05]. Reading from texture memory is often the main consumer of bandwidth [AMH02]. Thus *texture compression* is widely used in graphics systems. S3TC (S3 Texture Compression) is the most common texture compression scheme, including compression methods for the alpha channel and color channel of texture images. Our major contribution is to propose a novel alpha compression scheme, which can substitute for that of S3TC/DXT5. Moreover our work gives an inspiration to new texture compression data formats, from which innovative texture compression standard could derive.

2. Previous Work

S3TC is a group of related texture compression algorithms originally developed by S3 Graphics, Ltd [INH99]. It consists of 5 different variations, named DXT1 through DXT5, among which DXT2 and DXT4 are rarely used. DXT1 only handles RGB channel. A 4×4 pixel block (called a *tile*) is compressed to 64 bits, consisting of two RGB565 colors c_0, c_1 and 16 2-bit indices. If the first color c_0 as a 16-bit unsigned integer is greater than the second color c_1, two other colors are calculated, $c_2 = (2c_0 + c_1)/3$ and $c_3 = (c_0 + 2c_1)/3$. Otherwise $c_2 = (c_0 + c_1)/2$ and c_3 is transparent black. The indices decide each pixel's color value. DXT3 and DXT5 convert 16 input pixels into 128 bits, consisting of 64 bits of alpha data followed by 64 bits of color data. The color data are encoded the same way as DXT1, except that the four color version of DXT1 algorithm is always used.

DXT3 contains a 4-bit alpha value per pixel. DXT5 stores two 8-bit alpha values $_0$, $_1$ and 16 3-bit indices. If $_0 > _1$, six other alpha values are calculated, such that $_2 = (6 _0 + _1)/7$, $_3 = (5 _0 + 2 _1)/7$, $_4 = (4 _0 + 3 _1)/7$, $_5 = (3 _0 + 4 _1)/7$, $_6 = (2 _0 + 5 _1)/7$, and $_7 = (_0 + 6 _1)/7$. Otherwise if $_0 \leq _1$, four other alpha values are calculated such that $_2 = (4 _0 + _1)/5$, $_3 = (3 _0 + 2 _1)/5$, $_4 = (2 _0 + 3 _1)/5$, and $_5 = (_0 + 4 _1)/5$ with $_6 = 0$ and $_7 = 255$. The indices determine the alpha value for each pixel. Because DXT5 uses an interpolated alpha scheme, it generally produces superior results for alpha (transparency) gradients than DXT3 and is considered the most flexible general purpose compression codec.

† Supported in part by National High Technology Research and Development 863 Program (No.2008AA010901), National Sci&Tech Major Project (No.2009ZX01028-002-003, 2009ZX01029-001-003), and National Natural Science Foundation (No.60736012, 60921002, 61070025) of China. Email: jiangyifei01@gmail.com.

3. Enhanced Alpha Compression

S3TC's fixed-rate data compression coupled with the single memory access made it ideally suited for use in compressing textures in hardware accelerated 3D computer graphics. Its subsequent inclusion in Microsoft's DirextX led to widespread adoption of the technology and made it the de facto standard. More than 10 years have passed since Iourcha et al. [INH99] presented the compression algorithms. However it seems that no other texture compression scheme could challenge the supremacy of S3TC.

03	03	03	7b
03	03	03	7b
80	80	80	7b
80	80	80	7b

03	03	03	80
03	03	03	80
80	80	80	80
80	80	80	80

Figure 1: *An illustration of DXT5 alpha compression. Left: original alpha values. Right: alpha values calculated through linear interpolation.*

One interesting aspect of DXT5 is that the two alpha values are interpolated the same way as in the color channel, without taking the inherent properties of alpha data into account. Consider that if there are only 3 different alpha values in a tile and the derived data ($_2-$ $_7$) cannot cover the third value, errors and unnecessary computations were introduced. Figure 1 shows an illustration.

3.1. Motivation of EAC

Enhanced Alpha Compression (EAC) is motivated by the essential differences between transparency and color, mainly including the following two aspects:

- Transparency and color are two different physical properties of materials. Both of them are related to lighting but different in essence. Transparency embodies the light absorption, and is defined as the ratio of the absorbed to the incident power. As the result of light reflection, color is co-determined by the material and the ambient light.
- Transparency and color play different roles in computer graphics. It is no exaggeration to say that the main task of GPU is to compute each pixel's color. Colors are computed on a pixel-by-pixel (even sub-pixel) basis, based on various sophisticated shading models and theories. However alpha values are usually assigned per object, and alpha channel is often used to describe the object's outline.

3.2. Data Formats of EAC

The above analysis reveals the fundamental fact the diversity of alpha values is generally not as rich as that of color values, which conforms to our daily experience. Obviously DXT5 neglected this simple but important fact. Based on the perception of the essential differences, EAC defines new alpha data formats as shown in Figure 2. There are 3 possible formats of the compressed 64-bit alpha block, distinguished by the relative values of the first two 8-bit data.

α_0	α_0	*00*	*00*	$(00)_2 \times 16$
α_0	α_1	$[\alpha_2]$	$[\alpha_3]$	2×16 *indices*
α_{max}	α_{min}	3×16 *indices*		

Figure 2: *EAC alpha data formats. Top to bottom: format 1, format 2 and format 3.*

The first two bytes are interpreted as two unsigned integers. If the two integers are equal (format 1), all the 16 pixels have the same alpha values. The rest 48 bits are zeros.

If the first integer is smaller than the second one (format 2), the following two bytes are another two base alpha values $_2$ and $_3$. The last 32 bits are 16 2-bit indices. One 2-bit index can decide a value from $_0$, $_1$, $_2$, and $_3$. $_2$ and $_3$ are not requisite. If they are present, the four base values make an increasing sequence, such that $_0 <$ $_1 <$ $_2 <$ $_3$. Otherwise, zeros take the place of $_2$ and (or) $_3$.

If the first integer is greater than the second one (format 3), 8 base alpha values are calculated, such that $_0 =$ $_{min}$, $_7 =$ $_{max}$, $_1 = (6$ $_0 +$ $_7)/7$, $_2 = (5$ $_0 + 2$ $_7)/7$, $_3 = (4$ $_0 + 3$ $_7)/7$, $_4 = (3$ $_0 + 4$ $_7)/7$, $_5 = (2$ $_0 + 5$ $_7)/7$, $_6 = ($ $_0 + 6$ $_7)/7$. The rest 48 bits are 16 3-bit indices, used to decide each pixel's alpha value.

3.3. Compression and Decompression

Because of the asymmetry of texture access [Fen03], the compression can be implemented by software, and decompression should be fast and inexpensive to implement in hardware.

The compression of EAC employs two simple clustering control strategies besides linear interpolation method. If two alpha values are equal or their difference is below a certain threshold, such that $|$ $_i -$ $_k| = 0$ or $|$ $_i -$ $_k| \leq \Delta$, they belong to the same cluster. Here the control factor $\Delta = ($ $_{max} -$ $_{min})/14$ is essentially the radius of cluster. If the number of clusters are not greater than 4, alpha channel is compressed according to data format 1 or format 2. Otherwise interpolation is adopted and alpha channel is compressed according to data format 3.

EAC is targeted for hardware implementation. Figure 3 illustrates a possible hardware diagram for an EAC decompressor. When receiving a 64-bit compressed block, EAC firstly gets the data in the first two bytes, $_0$ and $_1$. The signal $flag$ is set depending on the relative values of $_0$ and $_1$. If $_0 \leq\ _1$, $flag = 0$ (mode 1). If $_0 >\ _1$, $flag = 1$ (mode 2). In mode 2 linear interpolation is employed to calculate the 8 base values, while in mode 1 two other base values are in the 3rd and 4th bytes of the compressed block. Finally the indices are consulted to determine each pixel's alpha value.

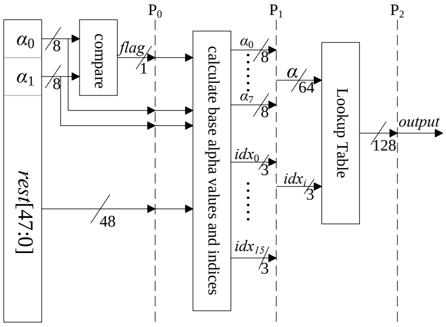

Figure 3: *An EAC decompressor structure. Decompression is completed in three cycles, the dash lines indicate registers.*

4. Experiments and Results

Both EAC and the alpha compression of DXT5 (DXT5AC) are implemented. The compression is implemented in C++ language and the decompression is implemented in Verilog.

4.1. Compression Quality

Transparency cannot be observed directly, and there are no such alpha data that are widely accepted as benchmarks. So we choose the Kodak image suite. The images are transformed to YCoCg-R color space [MS03], and the Y channel data are used for test. PSNR is used to measure quality. Final decompressed luma data are stored as grayscale images to facilitate observation.

Figure 4 shows the monochromatic images. As can be seen, the luminance changes more sharply in DXT5AC than in EAC, and the visual quality of EAC is better than that of DXT5AC. The average PSNR score of 24 grayscale images is increased by about 0.2 dB. The most significant increase is 0.667 dB from kodim08. The most subtle one is 0.007 dB from kodim02. Table 1 shows the detailed scores.

4.2. Synthesis Results

Decompressors of EAC and DXT5AC are synthesized with Synopsys Design Compiler on a 0.13 μm library by Semiconductor Manufacturing International Corporation (SMIC). Table 2 summarizes the synthesis reports.

When the clock period is set to 4ns, DXT5AC has 8 violating paths. Report shows that these paths start from registers at P_0 and terminate at registers labeled P_1 (see Figure 3).

Figure 4: *Top to bottom: enlargements of kodim02, 08 and 18 (partial). Arrows point to distinguishable parts.*

DXT5AC	EAC	Δ	DXT5AC	EAC	Δ
40.144	40.338	0.193	37.851	38.049	0.198
46.893	46.900	0.007	42.045	42.156	0.112
47.472	47.510	0.038	45.404	45.686	0.283
46.110	46.220	0.110	45.517	45.563	0.046
39.190	39.319	0.129	44.529	44.790	0.261
41.936	42.015	0.079	41.203	41.519	0.315
44.591	44.728	0.137	42.035	42.419	0.383
37.791	38.458	0.667	44.260	44.481	0.221
45.031	45.180	0.148	41.855	41.920	0.065
45.202	45.481	0.279	44.054	44.211	0.157
42.878	42.956	0.078	46.773	46.806	0.033
46.215	46.409	0.195	40.544	41.073	0.529
Average increase = 0.194 dB					

Table 1: *PSNR scores and the increments. Left: kodim01 to kodim12. Right: kodim13 to kodim24. Unit is dB.*

EAC does not have violating paths even with the clock period set to 1ns. Compared to DXT5AC, the cell area of EAC shows great drops. Under three different clock period settings, the total area is reduced by 25.8%, 26.8% and 24.2% respectively. The combinational area respectively decreases 42.7%, 43.1% and 41.2%.

4.3. Power Consumption

The two designs are re-simulated with the RTL codes replaced by gate-level netlists, generated with the clock period set to 4ns, to capture VCD (value change dump) files.

Name		DXT5AC	EAC	DXT5AC	EAC	DXT5AC	EAC
Levels of logic		30	5	27	6	25	4
Critical path length		3.77	2.15	3.45	1.67	3.4	0.77
Critical path slack		-0.01	1.62	-1.65	0.11	-2.62	0
Critical path clk period		4	4	2	2	1	1
Total negative slack		-0.08	0	-46.96	0	-113.43	0
No. of violating paths		8	0	46	0	176	0
Total area (μm^2)		41701.32	30939.42	43844.53	32083.86	45361.44	34385.22
	Comb	27087.86	15526.93	29191.54	16619.35	29882.37	17405.89
	Noncomb	14613.46	15412.49	14652.99	15464.51	15479.07	16979.33
Overall compile time		215.25	30.48	289.06	42.77	306.09	62.75

Table 2: *Design Compiler synthesis reports. Two designs are synthesized 3 times with the clock period set to 4ns, 2ns and 1ns.*

The compressed luma data of Kodak images are used as testbenches and 24 VCD files are captured for EAC and DXT5AC respectively. Synopsys Prime Power reads the VCD files and calculates the power consumption. Table 3 lists the detailed power consumption of DXT5AC and EAC. Left side shows the register power. Although DXT5AC and EAC have almost the same number of registers (310 registers in DXT5AC and 311 in EAC), the average register power is reduced by 13.44%. The right side gives the total power. The average total power is significantly lowered by 24.56%.

5. Conclusion

A novel alpha compression scheme, EAC, was presented and implemented based on the perception of the essential differences between transparency and color. Since EAC is superior in terms of visual quality, cell area, power consumption, etc, it can substitute for DXT5 alpha compression, which will significantly influence the modern graphics hardware.

Further improvements can be achieved when applying EAC. Sophisticated clustering algorithms, such as K-means, can be adopted to get higher PSNR scores. Clock-gating technique can be used to further reduce power consumption, since some registers can be disabled in EAC decompression mode 1. Moreover, in EAC data format 1 and format 2 (when $_3$ is absent), some bits are left unoccupied. The free bits can be utilized by color channel to further improve the color channel's compression quality, from which an innovative texture compression standard could derive.

Register Power			Total Power		
DXT5AC	EAC	Ratio	DXT5AC	EAC	Ratio
6.023e-4	5.182e-4	13.96	1.023e-3	7.553e-4	26.17
5.606e-4	4.867e-4	13.18	9.046e-4	6.889e-4	23.84
5.515e-4	4.790e-4	13.15	8.761e-4	6.771e-4	22.71
5.822e-4	5.038e-4	13.47	9.633e-4	7.235e-4	24.89
6.108e-4	5.252e-4	14.01	1.042e-3	7.659e-4	26.50
5.739e-4	4.963e-4	13.52	9.426e-4	7.108e-4	24.59
5.618e-4	4.882e-4	13.10	9.146e-4	7.000e-4	23.46
6.087e-4	5.229e-4	14.10	1.043e-3	7.665e-4	26.51
5.598e-4	4.887e-4	12.70	8.947e-4	6.912e-4	22.75
5.671e-4	4.929e-4	13.08	9.284e-4	7.054e-4	24.02
5.710e-4	4.927e-4	13.71	9.377e-4	7.060e-4	24.71
5.507e-4	4.805e-4	12.75	8.820e-4	6.813e-4	22.76
6.153e-4	5.286e-4	14.09	1.057e-3	7.742e-4	26.75
5.971e-4	5.148e-4	13.78	1.006e-3	7.460e-4	25.84
5.594e-4	4.841e-4	13.46	9.001e-4	6.876e-4	23.61
5.569e-4	4.840e-4	13.09	8.989e-4	6.882e-4	23.44
5.805e-4	5.029e-4	13.37	9.562e-4	7.201e-4	24.69
6.055e-4	5.226e-4	13.69	1.019e-3	7.553e-4	25.88
5.812e-4	5.042e-4	13.25	9.586e-4	7.247e-4	24.40
5.160e-4	4.490e-4	12.98	7.954e-4	6.231e-4	21.66
5.715e-4	4.984e-4	12.79	9.244e-4	7.074e-4	23.47
5.871e-4	5.083e-4	13.42	9.753e-4	7.315e-4	25.00
5.629e-4	4.874e-4	13.41	9.148e-4	6.970e-4	23.81
5.892e-4	5.074e-4	13.88	9.897e-4	7.346e-4	25.78
5.760e-4	4.986e-4	13.44	9.478e-4	7.151e-4	24.56

Table 3: *Power consumption reported by Prime Power. Top to bottom: kodim01 to kodim24. Data are presented in scientific notation, unit is Watt. The last row gives average values.*

References

[AMH02] AKENINE-MÖLLER T., HAINES E.: *Real-Time Rendering*, Second ed. A. K. Peters, Wellesley, Massachusetts, 2002, ch. 15, p. 689. 1

[AMHH08] AKENINE-MÖLLER T., HAINES E., HOFFMAN N.: *Real-Time Rendering*, Third ed. A. K. Peters, Wellesley, Massachusetts, 2008, ch. 18, pp. 841–842. 1

[Fen03] FENNEY S.: Texture compression using low-frequency signal modulation. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware* (2003), Eurographics Association, pp. 84–91. 2

[INH99] IOURCHA K. I., NAYAK K. S., HONG Z.: System and method for fixed-rate block-based image compression with inferred pixel values. *US Patent 5,956,431*, Sept. 1999. 1, 2

[MS03] MALVAR H., SULLIVAN G.: YCoCg-R: A color space with RGB reversibility and low dynamic range. JVT-document JVT-I014, JVT. 3

[Owe05] OWENS J.: Streaming Architectures and Technology Trends. In *GPU Gems2* (2005), Addison-Wesley, pp. 457–470. 1