

PACKMAN: Texture Compression for Mobile Phones

Jacob Ström
Ericsson Research

Tomas Akenine-Möller
Lund University / Ericsson Mobile Platforms

1 Introduction

We present a new lossy texture compression (TC) scheme, targeted for mobile devices, which compresses 2×4 pixel blocks into 32 bits. Compared to our POOMA TC work [Akenine-Möller and Ström 2003], peak signal to noise ratio (PSNR) is improved by 0.6 dB on average, the bit rate is reduced from 5.33 bits per pixel (bpp) to 4 bpp, and we avoid 2×3 blocks that are awkward for hardware implementation.

Most mobile devices, such as phones and PDAs, have very limited available memory bandwidth, e.g., 16 or 32 bits/cycle. For 3D graphics applications, TC is therefore a fundamental technique for reducing bandwidth usage. Most TC schemes compress each block of pixels individually. In general it is simpler to compress larger blocks efficiently, and therefore the most common block size is 4×4 pixels compressed into 64 bits (S3TC, DXTC). If a compressed block occupies the same number of bits as the bus width (e.g., 32 bits), then pipeline stalls can be avoided [Akenine-Möller and Ström 2003], which simplifies hardware implementation. For more information on different TC schemes, we refer to Fenney's [2003] previous work section.

2 Algorithm

The image is divided into 2×4 pixel blocks with each block represented by 32 bits. We take the unconventional approach of using only a single color per block, where twelve bits represent the "base color" quantized to four bits per R, G, and B. The remaining 20 bits modulate the luminance. For each pixel, the base color is additively modified with a constant taken from a small table with four entries. The same constant is added to all color components. This requires a *pixel index* of two bits per pixel to specify which of the four values should be used. The remaining four bits are spent on a *table index*, that specifies which table is used for the entire block. Decompressing a single pixel works as follows:

- I. The base color of the block is converted from 12 to 24 bits. As an example, $RGB=(0, 1, 15)$ is converted to $(0, 17, 255)$.
- II. The 4-entry table is located using the table index. For instance, a table index of 0 means that one of the following modifiers should be used: $\{-8, -2, 2, 8\}$ (see table below).
- III. The final color is computed by additively modifying the 24-bit base color with the value from the 4-entry table that corresponds to the pixel index. E.g., if the pixel index is 1, the modifier is -2 , and the final color is $(0, 17, 255) + (-2, -2, -2) = (0, 15, 253)$, where values are clamped to $[0, 255]$.

The tables associated with the table indices are shown below. Tables 8–15 equal tables 0–7 scaled by a factor of two. The table values were created by starting with random values and then optimizing them by minimizing the error for a test image (not Figure 1).

table index	0	1	2	3	4	5	6	7
	-8	-12	-31	-34	-50	-47	-80	-127
	-2	-4	-6	-12	-8	-19	-28	-42
	2	4	6	12	8	19	28	42
	8	12	31	34	50	47	80	127

Encoding can be done by exhaustively searching all parameters for the smallest mean square error. This takes about one minute for a 64×64 texture on a 1.2 GHz PC. Using the average color as the base color and exhaustively searching the other parameters is much quicker, taking only about 30 milliseconds.

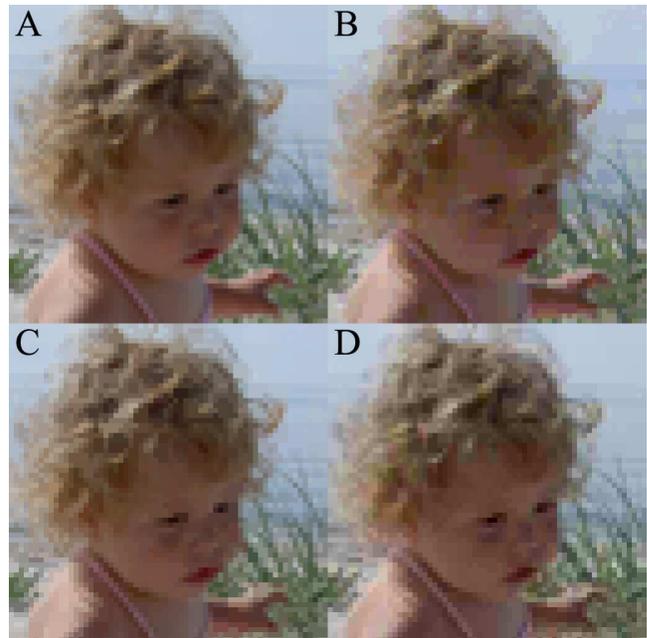


Figure 1: A: original, B: POOMA, 32.02 dB, 5.33 bpp, C: S3TC, 33.76 dB, 4 bpp, D: Proposed scheme, 33.04 dB, 4 bpp.

3 Results

An image compressed with the proposed scheme is shown in Figure 1 D, where the exhaustive search for the base color was used.

Decompression hardware for our scheme is tiny, requiring only three 9-bit adders, a multiplexor, a small lookup, clamping, and little else. Comparing to POOMA, the address calculation is simpler, since no division by three is required. Also, the bit rate and the quality in terms of PSNR is better. Comparing to S3TC, our decompression is simpler, since it avoids multiplication with $1/3$ or $2/3$. A block fits in 32 bits instead of 64 which is better for small bus widths. The bit rate is the same (4 bpp), but our PSNR is worse and smooth transitions between two non-gray colors are not as smooth.

Averaging over a small number of images, our PSNR score is 0.6 dB better than POOMA, and 0.8 dB worse than S3TC. Yet, visually, the quality is often closer to S3TC. This is so since our scheme spends more bits on the luminance, to which the human visual system is more sensitive than the chrominance. Comparing PSNR for the luminance only, we outperform S3TC with 0.6 dB on average, and POOMA with 1.3 dB.

This work is ongoing; we are currently investigating faster compression and error measures that are better targeted for the eye.

References

- AKENINE-MÖLLER, T., AND STRÖM, J. 2003. Graphics for the Masses: A Hardware Rasterization Architecture for Mobile Phones. *ACM Transactions on Graphics*, 22, 3, 801–808.
- FENNEY, S. 2003. Texture Compression using Low-Frequency Signal Modulation. In *Graphics Hardware*, ACM SIGGRAPH/Eurographics, 84–91.