JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 10, No. 2, June 2016, pp. 68-73

A Hybrid Texture Coding Method for Fast Texture Mapping

Li Cui
Department of Computer & Software, Hanyang University, Seoul, Korea and Department of Computer Science & Technology, College of Science & Engineering, Yanbian University, Yanji, China
Lcui2000@gmail.com

Hyungyu Kim
Department of Electronics & Computer Engineering, Hanyang University, Seoul, Korea
hyungyukimkr@gmail.com

Euee S. Jang*
Division of Computer Science & Engineering, Hanyang University, Seoul, Korea
esjang@hanyang.ac.kr

Abstract
An efficient texture compression method is proposed based on a block matching process between the current block and the previously encoded blocks. Texture mapping is widely used to improve the quality of rendering results in real-time applications. For fast texture mapping, it is important to find an optimal trade-off between compression efficiency and computational complexity. Low-complexity methods (e.g., ETC1 and DXT1) have often been adopted in real-time rendering applications because conventional compression methods (e.g., JPEG) achieve a high compression ratio at the cost of high complexity. We propose a block matching-based compression method that can achieve a higher compression ratio than ETC1 and DXT1 while maintaining computational complexity lower than that of JPEG. Through a comparison between the proposed method and existing compression methods, we confirm our expectations on the performance of the proposed method.

Category: Human computing

Keywords: Block matching; Texture compression; Spatial similarity

I. INTRODUCTION

Texture compression has been widely used for 3D graphics applications in order to reduce the size of surface texture images at the cost of rendering efficiency [1]. Several compression methods are currently used for texture compression. Among them, JPEG is widely used for compression of still images, including textural images. However, the lack of support for random access on the compressed image during texture mapping [2] is a well-known drawback of JPEG. To support random-access rendering, several methods have been proposed that split an initial image into uniform-size blocks and perform compression for each block separately. In PACKMAN texture compression (i.e., Ericsson Texture Compression [ETC]), a texture image is split into a number of 2×4 pixel

Open Access http://dx.doi.org/10.5626/JCSE.2016.10.2.68 http://jcse.kiise.org

Received 15 October 2015; **Revised** 27 April 2016; **Accepted** 01 June 2016
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

blocks and compressed into 32 bits per block, which simplifies hardware implementation [3]. Strom and Akenine-Moller [4] proposed an extension of ETC called ETC1, which splits a texture image into 4×4 pixel blocks. A new compression method, called ETC2, adds three modes to ETC that handle chrominance edges better than ETC1 [5]. ETC1 and ETC2 provide better image quality than the conventional methods used in DirectX, OpenGL, and iOS-based systems with relatively low complexity such as S3TC/DXT1 [6], PVRTC [7] and BPTC [8], in exchange for worse compression efficiency. Using ETC1 as the demonstration codec, Strom and Wennersten [9] decreased transmission time over the network through further packing of compressed textures. Two limitations of this method still remain: high computational complexity, and the lack of ability to trade quality for bit rate. In summary, it remains a challenging task to find an optimal trade-off between compression efficiency and computational complexity for fast rendering of texture images.

The block matching algorithm (BMA) is the basis of a template matching scheme that has long been used in several fields involving image and video processing. The BMA compares each of a number of macroblocks with a corresponding block within a given search range to assess temporal redundancy. Several BMAs have been proposed, which are based on different search ranges. Full-search BMA can obtain high compression efficiency at the cost of computational complexity [10]. The three-step and diamond BMAs reduce the processing time for the encoding process through effectively reducing the search range compared with the full search algorithm [11, 12]. A large search range can achieve a higher compression ratio, but it also results in an increase in encoding complexity. Determining the search range of BMA is a critical problem in video coding. Although BMA is often used to improve the inter-frame coding performance in video compression, it can also be applied for intra-frame coding. Intra-BMA, called intra-block copy (IntraBC), has recently been adopted into an extension of High Efficiency Video Coding (HEVC) for screen content coding [13-15]. The search range of IntraBC is limited to the left coding tree unit (CTU) and the reconstructed region of the current CTU.

In this paper, we present an efficient texture compression method based on the intra-picture block matching method to improve compression efficiency while preserving low complexity. High decoding speed for texture compression is the most significant requirement, while encoding speed is not essential, which is different from the case with video coding. Therefore, the proposed method selects all of the previously encoded blocks as the search objectives of BMA to achieve an optimal solution. The similarity of each block is measured through the block matching process. According to the extent of similarity, a block is encoded using a coordinate of the corresponding reference block; otherwise, it is encoded by the conventional compression method. Moreover, an acceptable threshold is used to restrict the distortion of each matched block in the texture image in order to adjust the compression ratio. This is a fundamental difference of the proposed method compared with BMA as used in video coding. By exploiting spatial redundancy, the proposed method obtains high compression efficiency for texture images with high degrees of similarity between blocks. Even in the worst-case scenario, where all blocks are encoded using the conventional compression method, only a small overhead (i.e., one-bit flag per block) is added to the conventional method. The computational complexity of the proposed method is slightly higher than those of the fast compression methods (e.g., ETC1), but still sufficiently lower than that of JPEG.

The rest of this paper is organized as follows: in Section II, we introduce the proposed method in detail, including the block matching process along with the decoding process. In Section III, the experimental conditions are described and the experimental results are analyzed through comparison with the results of other methods. Finally, we provide concluding remarks and suggest future research directions in Section IV.

II. HYBRID TEXTURE CODING METHOD

To achieve a reasonable trade-off between compression efficiency and computational complexity, we propose a compression method to reuse previously encoded blocks through a block matching process.

A. Encoding Process

In the proposed method, the type of each block is determined through the block matching process based on an acceptable threshold (*th*). There are two possible block types: 1) an independent block (IB), which is encoded using the conventional compression method, and 2) a repetition block (RB), which is replaced by one of the IBs. Fig. 1 shows the flow chart of the block matching process for each block. The smallest distortion between the current block and each of the previously encoded blocks is computed. Then, the smallest distortion is compared with the specified threshold. If the smallest distortion owned by the block (b_j) is not larger than the threshold, b_j is defined as the IB and the current block (b_c) is determined to be an RB. This process is performed block by block until the types of all blocks are determined. Finally, the type of each block, the data of IBs, and the coordinates of RBs are encoded.

Compression efficiency can be improved by replacing an RB with a coordinate related to another IB. The compression ratio (R) of the proposed method is correlated with the proportion of the blocks encoded by the conven-

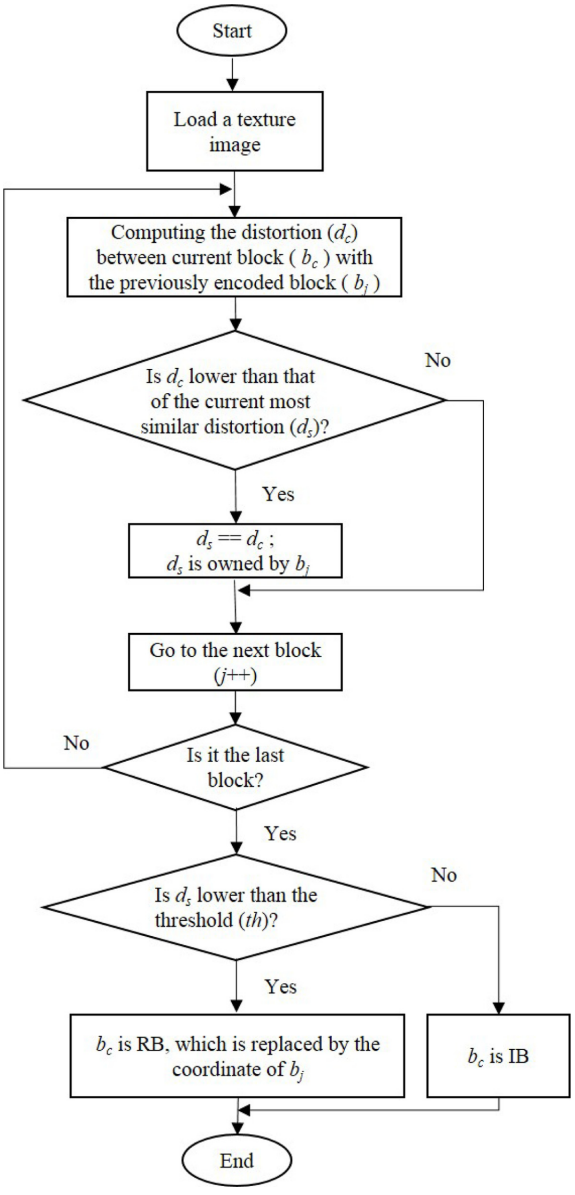

Fig. 1. Flow chart of the block-matching process. IB: independent block, RB: repetition block.

tional method (p_{con}), as follows:

$$R = \frac{b_{con}}{1 + p_{con} b_{con} + (1 - p_{con})(\log_2 M + \log_2 N)} \quad (1)$$

where b_{con} is the number of bits per block according to the conventional method and the resolution of the image is $M \times N$. The range of the bit length that represents the coordinate of the IB can be computed by M and N. For example, $\log_2 512 + \log_2 512 = 18$ bits for the 512×512 24-bit RGB image. Also, b_{con} is 64 bits when ETC1 is used as the conventional compression method. In this situation, R converges to 64/19 as more blocks are encoded using the proposed method (i.e., as p_{con} approaches zero). In summary, as more blocks are encoded by the proposed method, a higher compression ratio can be achieved.

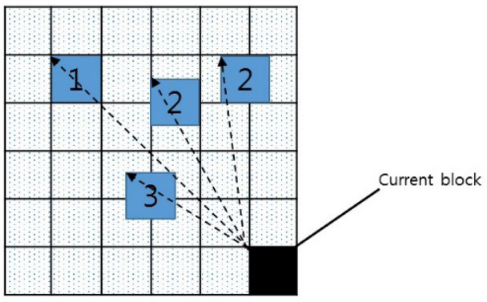

Fig. 2. Three possible cases based on the position of the matched block.

B. Decoding Process

In exchange for improved compression efficiency, computational complexity may be increased for the following reason. The decoding time is increased because all of the reference blocks of an RB should be decoded. Because the coordinate is given on the pixel level, the reference block may overlap with more than one conventionally-coded blocks during the decoding process, as shown in Fig. 2. The probability of the three cases depicted in Fig. 2 can affect the decoding efficiency. A linear relationship exists between the decoding time of the proposed method (T_p) and the decoding time of the conventional compression method (T_c), as follows.

$$\begin{aligned} T_c &= M_t + B_t \\ k &= p_1 \times 1 + p_2 \times 2 + p_3 \times 4 \\ T_p &= M_f + (p_c + k - p_c \times k) \times T_c \end{aligned} \quad (2)$$

where M_t represents the time of memory access for blocks, B_t is the time consumed in decoding blocks, p_1, p_2, and p_3 are the probabilities of three cases of memory access, p_c is the probability of the blocks encoded by the conventional compression method, and M_f represents the time needed for memory access given the type value of the blocks. T_p is slightly greater than T_c according to the proportions of the three cases, which in the worst-case scenario is close to four times more than T_c.

C. Bitstream Structure

We implement a fast parsing process for the proposed method to restore random access capability. Fig. 3 shows the structure of the compressed bitstream in the proposed method. One bit is required to indicate the block type information for each block. During the decoding process, the block type is read first. If the type of the current block is IB, the data of the IB at the corresponding position is read directly. If the type of the current block is RB, the

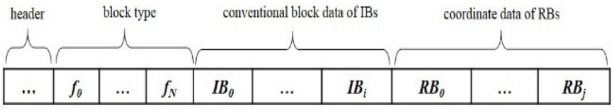

Fig. 3. Bitstream structure of hybrid texture coding.

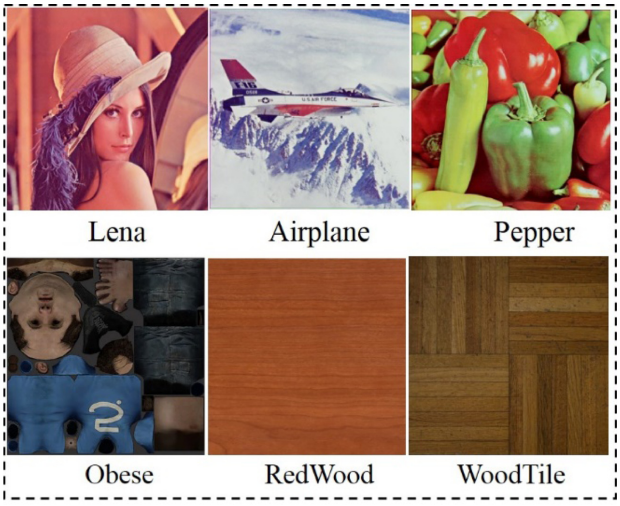

Fig. 4. Test images.

coordinate value of RB at the corresponding position is read. The corresponding block(s) are read from the conventional block data according to the coordinate value. The number of blocks (*N*) in a texture image equals the sum of the number of IBs (*i*) and the number of RBs (*j*), as shown in Fig. 3.

III. EXPERIMENTAL RESULTS

A. Test Condition

To investigate the performance of the proposed method, we used ETC1 and DXT1 as the conventional compression methods. Ten 512×512 24-bit RGB color images were used in our experiments, which include conventional images (e.g., Lena, Airplane, and Pepper) as well as texture images (e.g., Obese, RedWood, and WoodTile), as shown in Fig. 4. We specified six constants as thresholds for the block matching process in order to observe performance change based on different mean squared error (MSE) value.

B. Results

Our first experiment compared the RD performance of the proposed method with that of the conventional methods (i.e., DXT1 and ETC1) for the evaluation of compression efficiency. Fig. 5 shows the comparison results

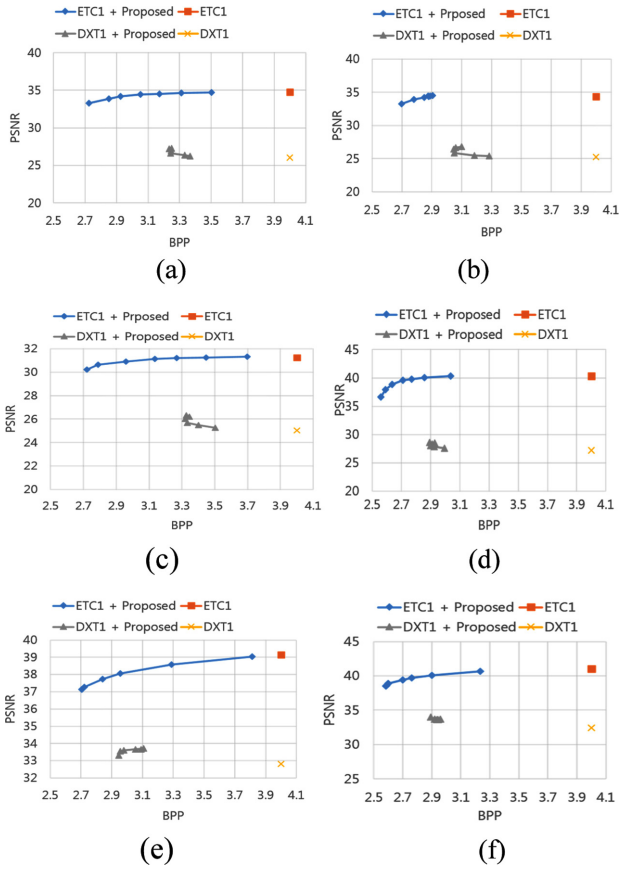

Fig. 5. Repetition block performance comparison between the proposed method and other conventional methods. (a) Lena, (b) Airplane, (c) Pepper, (d) Obese, (e) RedWood, and (f) WoodTile.

for the six test images. The proposed method achieves a better compression ratio than the conventional methods while preserving comparable image quality. The quality of the images that contain a greater number of similar texture areas (e.g., Airplane, RedWood and WoodTile), is significantly worse than that of other images near the smaller threshold values. This finding indicates that the quality of a simple image is more sensitive than that of a complex image using the proposed method and a small threshold.

The second experiment was done to evaluate the decoding performance; the results shown in Table 1 are based on the test images being decoded 300 times by the corresponding methods. The decoding performance of JPEG was measured based on a quality factor (QF), and the threshold (*th*) is the measuring basis for the proposed method. The decoding time of JPEG is much longer than that of both the conventional methods and the proposed method, because JPEG has to decode an entire image in order to decode one region of that image in order to address the random access of texture data. Moreover, the results in Table 1 show that the decoding time of the proposed method is slightly greater than that of the conven-

Table 1. Comparison of the average decoding time (ms)

Method	Decoding time (ms)
JPEG	7.17×10^4
ETC1	1.80
DXT1	1.83
Proposed + ETC1	2.01
Proposed + DXT1	2.06

Table 2. Average color difference comparison of the rendered results for ObeseMale

Method	*Th*	PSNR	CIE94 (%)	RMS
ETC1		43.08	9.64	3.11
Proposed + ETC1	50	42.47	9.86	3.33
	100	41.82	10.12	3.58
	200	40.52	10.72	4.16
	400	35.87	12.67	7.26
	800	35.69	14.18	8.89
	2000	32.11	18.29	10.96
DXT1		30.41	26.73	13.61
Proposed + DXT1	50	31.16	18.80	12.53
	100	31.31	18.16	12.27
	200	31.46	16.81	12.09
	400	31.73	17.98	11.70
	800	31.91	17.52	11.47
	2000	31.98	18.03	11.36

tional methods, as described in Eq. (2). In addition to comparing the results of the proposed method, we found that the decoding time is increased in high-threshold cases. This demonstrates that when the proposed method is performed with a larger threshold, more blocks are coded as RBs, resulting in more reference blocks to be decoded.

The next experiment was done to evaluate the quality of the rendered model using a texture image. Conventionally, image quality evaluation has been conducted by comparing the PSNR performance between the reference and distorted images. Performance evaluation using final rendered images matches how humans perceive image distortions and has been reported as a texture compression method [15]. Thus, we rendered a 3D Model (ObeseMale) using the Obese texture image compressed by using ETC1, DXT1, and the proposed method. The viewpoint space for the single-object models in [15] was sampled using the quasi-random Sobol sequence generator, but identifying the expected viewer locations is left for future work. In our experiment, we fixed the Y-coordinate of the viewpoints as an acceptable value. In the XZ-plane, the viewpoint space was uniformly sampled into ten viewpoints to form a set of evaluation viewpoints. The rendered results using the compressed Obese test image were captured at each viewpoint, and they are compared each to a ground truth using the uncompressed Obese test image from the same viewpoint. The average PSNR, CIE94, and RMS performance based on ten viewpoints were measured, as shown in Table 2. The results show that the texture images in the rendering stage can achieve better quality than that of the texture images. This implies that the bitrate of the compression methods can be reduced while still maintaining rendered quality.

IV. CONCLUSION

This paper presents a hybrid texture coding method that utilizes the spatial similarity between previously encoded blocks and the current block based on a block matching process. The experimental results show that the proposed method can achieve a flexible bitrate based on a specified threshold and obtain a good trade-off between compression efficiency and computational complexity compared to conventional methods. We expect that the proposed method can be further improved by more efficient representation of the coordinates of the reused blocks within the bitstream structure.

REFERENCES

1. P. S. Heckbert, "Survey of texture mapping," *IEEE Computer Graphics and Applications,* vol. 6, no. 11, pp. 56-67, 1986.
2. G. K. Wallace, "The JPEG still picture compression standard," *IEEE Transactions on Consumer Electronics*, vol. 38, no. 1, pp. 18-34, 1992.
3. J. Strom and T. Akenine-Moller, "PACKMAN: texture compression for mobile phones," in *ACM SIGGRAPH Sketches*, Los Angeles, CA, 2004, p. 66.
4. J. Strom and T. Akenine-Moller, "iPACKMAN: high-quality, low-complexity texture compression for mobile phones," in *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, Los Angeles, CA, 2005, pp. 63-70.
5. J. Strom and M. Pettersson, "ETC2: texture compression using invalid combinations," in *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, San Diego, CA, 2007, pp. 49-54.
6. K. I. Iourcha, K. S. Nayak, and Z. Hong, "System and method for fixed-rate block-based image compression with inferred pixel values," US Patent 5,956,431, February, 1999.
7. S. Fenny, "Texture compression using low-frequency signal modulation," in *Proceedings of ACM SIGGRAPH/EUROGRAPHICS Conference on Graphics Hardware*, San Diego, CA, 2003, pp. 84-91.
8. OpenGL, "ARB_texture_compression_bptc," 2010; https://www.

opengl.org/registry/specs/ARB/texture_compression_bptc.txt.
9. J. Strom and P. Wennersten, "Lossless compression of already compressed textures," in *Proceedings of ACM SIGGRAPH Symposium on High Performance Graphics*, Vancouver, Canada, 2011, pp. 177-182.
10. Y. C. Lin and S. C. Tai, "Fast full-search block-matching algorithm for motion-compensated video compression," *IEEE Transactions on Communications*, vol. 45, no. 5, pp. 527-531, 1997.
11. X. Jing and L. P. Chau, "An efficient three-step search algorithm for block motion estimation," *IEEE Transactions on Multimedia*, vol. 6, no.3, pp. 435-438, 2004.
12. C. H. Cheung and L. M. Po, "Novel cross-diamond-hexagonal search algorithms for fast block motion estimation," *IEEE Transactions on Multimedia*, vol. 7, no. 1, pp. 16-22, 2005.
13. C. C. Chen, X. Xu, R. L. Liao, W. H. Peng, S. Liu, and S. Lei, "Screen content coding using non-square intra block copy for HEVC," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, Chengdu, China, 2014, pp. 1-6.
14. D. K. Kwon and M. Budagavi, "Fast intra block copy (IntraBC) search for HEVC screen content coding," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, Melbourne, Australia, 2014, pp. 9-12.
15. W. Griffin and M. Olano, "Evaluating texture compression masking effects using objective image quality assessment metrics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 8, pp. 970-979, 2015.

Li Cui

Li Cui received her B.S. degree from Yanbian University, Yanji, China in 2000 and the M.S. degree from Honam University, Korea in 2007. She is currently working toward a Ph.D. at Hanyang University, Seoul, Korea. She is also a lecturer in the Department of Computer Science and Technology, College of Science and Engineering, Yanbian University, Yanji, China. Her current research interests include MPEG Reconfigurable Media Coding, MPEG Reconfigurable Graphics Coding, and computer graphics.

Hyungyu Kim

Hyungyu Kim received his Ph.D. degree from Hanyang University, Seoul, Korea, in 2014. He is currently a post-doctorate researcher in the Division of Computer Science and Engineering of Hanyang University. He is interested in the MPEG Reconfigurable Media Coding framework: his main research topics are the bitstream syntax description and the bitstream parser generation methodology. He has received a Certificates of Appreciation from International Standards Organization and International Electro-technical Commission in 2011 for his contribution to the development of ISO/IEC 23001-4 international standard.

Euee S. Jang

Euee S. Jang received his B.S. degree from Jeonbuk National University, Korea and a Ph.D. from SUNY at Buffalo, NY, USA. He is a Professor in the Division of Computer Science & Engineering, Hanyang University, Seoul, Korea. His research interests include image/video coding, reconfigurable video coding, and computer graphics objects. He has authored more than 150 MPEG contribution papers, more than 30 journal or conference papers, 35 pending or accepted patents, and two book chapters. Dr. Jang has received three ISO/IEC Certificates of Appreciation for contributions to MPEG-4 development. He received the Presidential Award from the Korean Government for his contribution to MPEG standardization.