A LOW-COMPLEXITY AND LOSSLESS REFERENCE FRAME ENCODER ALGORITHM FOR VIDEO CODING

Dieison Silveira, Guilherme Povala, Lívia Amaral, Bruno Zatt, Luciano Agostini, Marcelo Porto

Group of Architectures and Integrated Circuits – GACI
Federal University of Pelotas – UFPel
Pelotas – RS – Brazil

ABSTRACT

This paper presents a lossless coding solution to reduce the large overhead of external memory communication during the motion estimation process in current video coders. Our solution is called Differential Reference Frame Coder (DRFC), and uses two techniques together to compress the reference frame: a differential coding based on a simplified intra-prediction process to reduce the spatial redundancy of the reference samples, and a simple VLC applied to differential coding residues. The proposed solution reaches an average compression rate higher than 45% for the evaluated HD 1080p video sequences. This is a lossless and low-complexity solution, and could easily be implemented in hardware.

Index Terms— Lossless reference frame compression, differential coding, memory bandwidth reduction, motion estimation, video coding.

1. INTRODUCTION

The recent multimedia innovations introduced an increasing demand for videos with better quality and higher resolutions. However, these videos require a large volume of data for representation, storage and eventual transmission. Furthermore, multimedia applications using digital videos are supported by mobile devices (such as smart phones, tablets, and digital cameras) that present severe limitations in terms of processing performance and battery capacity. Due to these limitations, video coding represents a key challenge in order to make multimedia support a feasible task for current systems.

The video coding process comprehends a variety of tools and techniques that were, and still are, being developed, so there is a great research activity in this area. Through such techniques, digital videos are represented with a much smaller volume of data, at the cost of minimal losses in visual human perception. The H.264/AVC [1] is the state-of-the-art video coding standard consolidated in the market and academy [2]. However, video-coding experts developed a new standard, the HEVC (High Efficiency Video Coding) [3] that achieves double the compression rate when compared to H.264/AVC, for the same video quality [3].

The video coder demands expressive external memory traffic to perform the coding, especially during the motion estimation (ME) process, which is responsible for major part of the memory communication. This occurs because the encoded frames are stored in external memory to be used as reference during the ME encoding process of future frames. As far as memory communication significantly impacts in energy consumption (up to 90% [4]), the coder project must consider it as a major system bottleneck.

Solutions to reduce memory accesses are based on two main approaches: (1) external memory bandwidth reduction through data reuse strategies using caches [5] and (2) bandwidth reduction by reference frames compression before they are stored in the external memory [6]-[11]. The main advantage of reference frame compression when compared to data reuse is that the former reduces the number of both reading and writing accesses, while the latter reduces the number of reading accesses only. Therefore, an efficient reference frame coding algorithm is desirable in order to promote meaningful external memory bandwidth reduction while avoiding increased computational cost.

This paper presents a lossless and low-complexity algorithm for the memory bandwidth reduction in video coding systems. This algorithm is called Differential Reference Frame Coder (DRFC). DRFC uses a two-step encoding, in the first step a differential coding between samples of the block is performed, and in the second step, the differential coding residue is substituted by optimized codes, and these codes may three different sizes: 4, 8 or 12 bits per sample. DRFC encoding process is performed without any information loss, providing exactly the same reference frames after the decoding processes.

This paper is organized as follows: Section 2 presents related works; Section 3 presents a digital videos statistical analysis, and DRFC algorithm; the achieved results and a comparison with related works are shown in Section 4; then, conclusion and future works are presented in Section 5.

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

2. RELATED WORKS

A number of works proposing lossy and lossless strategies to compress reference frames are found in the literature, such as [6], [7], [8], [9], [10], and [11]. These works are discussed in the following subsections 2.1 and 2.2.

2.1. Lossy Reference Frame Compression

Paper [6] applies an in-loop compression of reference frames. This paper uses a fixed-length compression algorithm for compressing reference frames. Its algorithm is called min-max scalar quantization (MMSQ) and it is based on a block scalar quantization scheme. This scheme saves from 25% to 37.5% of the memory transfer bandwidth. However, this algorithm introduces a quality loss from 0.043 dB to 0.159 dB.

Work [7] proposed a solution based on the MMSQ algorithm [6]. The improvement proposed by [7] reduces the losses of the MMSQ process by storing the errors, so that these errors can be returned to the correspondent blocks during the motion compensation process. This technique can achieve an average memory bandwidth reduction of about 23%, introducing 0.01dB quality loss.

The main drawback of lossy approaches is encoder/decoder drifting present in this approach. This causes the decoded video quality becomes lower than expected. Thus, a lossless algorithm is desired because it ensures the video quality during encoding.

2.2. Lossless Reference Frame Compression

In work [8], a lossless frame recompression is proposed to reduce memory bandwidth. The compression is performed on 8x8 partition sizes. Different scanning modes are used to calculate DPCM (Differential Pulse Code Modulation) values, and then variable length coding (VLC) is applied to these values to express them in fewer bits. To calculate DPCM values, six scanning modes have been considered. These techniques combined can achieve an average memory bandwidth reduction of about 50% to 60%, when processing high definition videos, however, only 10 frames of 5 sequences were used in experiments. This solution is not hardware-efficient, since a hardware implementation of such solution will demand a lot of hardware resources and incur in increased memory-encoder latency.

Paper [9] proposes an algorithm that uses a hierarchical minimum and difference (HMD) method to calculate the difference between pixels in an 8x8 block and to encode the differences using the Exp-Golomb coding. On the average, it achieves 34% of data reduction.

Work [10] proposes a compression algorithm to reduce the memory size for display devices. The algorithm uses a modified Hadamard transform (MHT) to correlate the pixels in a line and compresses the correlated coefficients using an adaptive Golomb-Rice coding method. This solution achieves, on the average, 31% of data reduction. This solution is not directly related to compression of reference frames, but can be used in this situation.

Work [11] proposed a solution context adaptive reference frame compressor that uses eight static Huffman tables. These tables are generated based on the statistical analysis of the samples distribution. Each one of these tables is specialized to a particular video characteristic. This solution reaches an average compression rate of about 31% for the evaluated video sequences.

In general, the related solutions demand increased computational cost or incur in undesirable quality losses, especially for reference frames. Thus, we propose the DRFC which is able to expressively reduce the external memory communication and to provide a lossless solution implementing a light-weight coding algorithm. Furthermore, the DRFC hardware architecture is easily implementable with low hardware cost.

3. STATISTICAL ANALYSIS AND THE DRFC

An analysis with many high-resolution video sequences was made to evaluate the behavior of the original and differential coded sample values distribution. The original samples are reconstructed samples obtained through the HM-12.0 reference software. The next subsection 3.1 shows the results of this analysis. Then, in subsection 3.2 we present the DRFC algorithm, in detail.

3.1. Differential Coding Analysis in the Video Coding

Fig. 1 presents a histogram for the extracted results from 12 well known HD1080p (1920x1080 pixels) video sequences that present different lighting and motion activity characteristics. The complete video sequences were used in the experiments, but only luminance samples were considered, since these samples are used in the ME process, at least for current video encoders that need to operate in real time.

As shown in Fig. 1(a) high occurrences happen along the whole distribution, this behavior causes that compression algorithms do not achieve high compression ratios. For this reason, the work [11] explores ways to split this distribution into smaller intervals, in order to concentrate the high occurrences in a few samples, and thus increase the compression rate. However, a simple way to change this behavior is to apply a differential coding between samples. This forces the residues generated by differential coding to be mostly zero, or very close to that. The histogram for this distribution can be seen in Fig. 1 (b).

Fig. 1 (b) shows residue samples histogram where can be observed a high concentration of values close to zero. This behavior is due to the fact that digital videos have high spatial redundancy; this means neighboring samples have very similar values. The differential coding is performed through the difference between two adjacent samples.

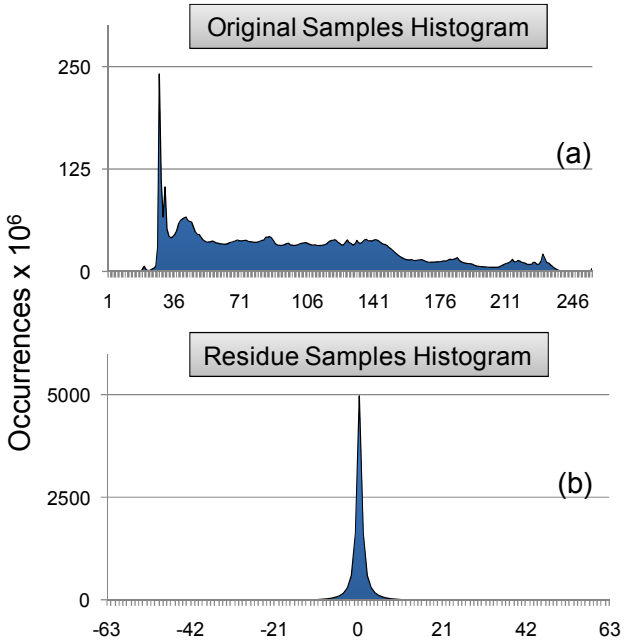

Fig. 1. (a) Reconstructed and (b) residue samples histogram for 12 HD 1080p video sequences.

When a differential coding is applied, spatial redundancy is eliminated resulting in a zero-centered distribution. In this case, 93% of the residues are in the range of -7 to 7.

3.2. DRFC Algorithm

The DRFC encoder receives 16x16 sample blocks from the video encoder, compresses these blocks and stores the result in the external memory. Additionally, when the video encoder requests a block from the external memory, this block must be fetched from memory and decoded by the DRFC. The differential coded samples are coded with a simple VLC. The scan order used in differential coding is performed through the difference between two adjacent samples, column by column. This process is conducted by three loops and can be seen in Fig 2 (*lines 3, 6, 7*).

The VLC used can generate three different code sizes: 8-bits size when the first sample of the block is coded; 4-bits size when the residue value is between "-7" and "7"; and 12-bits size when the residue value is outside this interval. The range "-7" to "7" was chosen because 93% of the residue values are in this interval. Therefore, this range consists of 15 different values, thus, a 4-bits code is sufficient to represent the range values.

The pseudo-code of DRFC is presented in Fig. 2. The DRFC encoding process starts with the fetch of a sample block. The first sample block is stored without any change, as can be seen in Fig. 2 (*line 2*), since this sample is used to start the process of decoding the block. The next samples to be coded compose the block first column (*lines 3, 4, 5*), in this step is performed differential coding between

```
// N == 16, 16x16 samples block
1. compressRecBlock(recBlock[N][N])
2.   assembler.append(recBlock[0][0])
3.   for (i=0; i< N-1; i++)
4.     encResidue = encode(recBlock[i][0], recBlock [i+1][0])
5.     assembler.append(encodedResidue)
6.   for (i=0; i< N; i++)
7.     for (j=0; j< N-1; j++)
8.       encodedResidue = encode (recBlock[i][j], recBlock[i][j+1])
9.       assembler.append(encodedResidue)
10.  return (assembler)
// encode(), calculates the residue and returns the sample coded
11. encode(sample, neighbor)
12.   residue = sample - neighbor
13.   switch (residue)
14.     case 0: encoded = 0000  //4 bits
15.     case 1: encoded = 0001  //4 bits
16.     case -1: encoded = 0010 //4 bits
17.     case 2: encoded = 0011  //4 bits
18.     .....                   //4 bits
28.     case -7: encoded = 1110 //4 bits
29.     default: encoded = 1111+ sample //12 bits
30.   return (encoded)
```

Fig. 2. DRFC algorithm

the current sample and the previous sample. It generates a residue that is coded according to its value, e.g., a residue value equal to "-5" (9 bits) when coded becomes "1011" (4 bits). If the value of the residue is less than "-7" or greater than "7", is assigned an exception code (4 bits) and the original sample value (8 bits) is attached to this code, generating a code with 12 bits size. The codes that are assigned to the residues can be seen in Fig. 2 (*lines 14-29*).

After the first-column samples were encoded, the differential coding process changes. Now, the differential coding is performed between the current sample and the previous column sample (*lines 6-9*), the residual coding follows the process previously presented. Since codes have a variable length (4, 8 or 12 bits), they are assembled in 32-bit words (this size can be modified to best fit to the external memory words) to be stored. The same process is applied for the next blocks of the frame.

The decoding phase consists in reading the coded block from the external memory and applying the reverse process of the encoding phase. The block is read as a list of words and the translation is performed using a greedy strategy, i.e., the decoder consumes the bits from the list, reading one word at a time. The first coded sample has 8-bits size, this sample is the original sample, and will start decoding phase. Thereafter, the read codes have 4-bits size, this code is between "0000" and "1110", and the residue value that corresponds to the code is obtained, and this residue and the previously decoded sample are added, following the reverse process of the encoding phase. If the read code is equal to "1111", the next 8 bits should be read, in this case it is not necessary to perform the samples sum, and these 8 bits compose the decoded sample. The decoding continues until the block is completely decoded.

4. COMPRESSION RESULTS AND COMPARISONS WITH RELATED WORKS

This section presents results achieved by the proposed solution and a comparison with related works. The first results are presented in Fig. 3, which shows the results obtained with the DRFC implementation, running each one of the 12 evaluated HD 1080p video sequences. As far as the DRFC compresses the reconstructed reference frames, the experiments were performed using the HM-12.0 with EPZS search algorithm and QPs (Quantization Parameter) 22, 27, 32, and 37. Fig. 3 brings the average results for the four considered QPs. The compression rates for each sequence indicate also the external memory bandwidth reduction achieved with our technique.

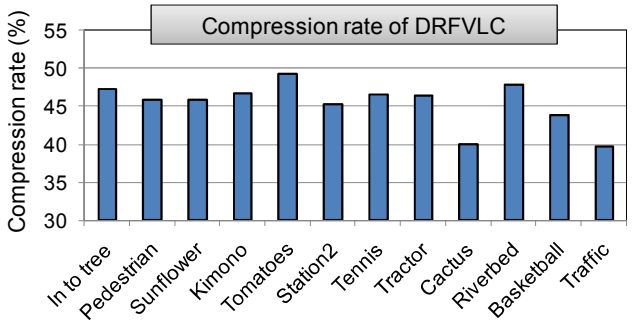

Fig. 3. Compression rate of DRFC.

Our solution achieves, on average, 45.37% of compression ratio with a 2.7% standard deviation, indicating that the DRFC achieves high compression rates with low variation regarding the used video. It is also important to note that the achieved compression rate impacts in the reduction of reading and writing accesses to external memory. It reflects directly in the encoder energy consumption reduction, since the communication with external memory has important impact on the total energy consumption of the system [4]. This is even more important when it comes to mobile devices, which have strict restrictions in energy consumption.

Table 1 presents a comparison between the DRFC results with related works found in the literature [6]-[11]. Three metrics were used in the comparison presented in Table 1: the average compression rate, the average quality degradation and the computational cost.

Table 1. Comparisons with related works.

Solution	Compression Rate	Δ PSNR	Computational cost
Our	45.37%	0.00	1 DC + 1 VLC
[6]	37.50%	-0.159	1 Quant.
[7]	23.00%	-0.01	1 Quant.
[8]	60.00%	0.00	6 DPCM + 6 VLC
[9]	34.08%	0.00	1 HMD + 1 VLC
[10]	31.40%	0.00	1 MHT + 1VLC
[11]	31.69%	0.00	1 VLC

As Table 1 presents, DRFC achieves better compression results than works [6], [7], [9], [10], and [11], even without introducing any quality loss. The works [6] and [7] present a lower computational cost if compared with our work, since those works only use a scalar quantization to compress the frames. However, [6] and [7] insert quality losses causing the encoder/decoder drifting, thus the decoded video quality becomes lower than expected. Papers [9], [10], and [11] use lossless compression schemes, however our work reaches 40% more compression rates for the same cost computational.

The comparison with [8] shows that our solution presents a lower compression rate, however, with a much lower computational cost. This related work uses six independent DPCMs, followed by six VLCs and a final decision to select the result with the lowest bit-rate. This high computational cost is inserted between the external memory and the encoder creating a new bottleneck in the whole system. This will cause an extra delay in a critical point of the system, negatively impacting the global video encoder throughput. Other point is that the solution proposed in [8] is not hardware-efficient, since a hardware implementation of such solution will demand large amount of hardware resources to implement dedicated compression data paths, internal memories, cache memory and logic to control the process. In turn, DRFC is a hardware-efficient algorithm, with high coding efficiency and low cost to be designed in hardware.

Summarizing, from the presented results and comparison, it is possible to conclude that our work presents a competitive compression rate, without introducing any quality loss. Our solution presents the lower computational cost among the lossless related works, and also can be easily hardware implemented.

5. CONCLUSION

This work presented a lossless and low computational cost algorithm for memory bandwidth reduction in video coding systems through reference frames compression. This solution uses a lossless differential coding and VLC method over blocks with 16x16 samples to reach the desired bandwidth reduction. The DRFC achieves an average memory bandwidth reduction of more than 45% on HD 1080p video sequences. This method is fully compliant with state-of-the-art coding standards, such as the H.264/AVC and the HEVC.

As future work, the DRFC will be described in VHDL and synthesized for FPGAs and ASIC using TSMC standard cells technology library. Also, the impacts in terms of hardware resources and power consumption for a hardware implementation of the DRFC will be evaluated.

6. REFERENCES

[1] ISO/IEC: International Organization for Standardization. ISO/IEC 14496-10 mpeg-4 part 10 - coding of audio-visual objects - part 10: Advanced video coding. Technical Report, 2012.

[2] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard", IEEE Transactions on Circuits and Systems for Video Technology, vol.13, pp.560-576, 2003.

[3] ITU-T Recommendation H.265: High Efficiency Video Coding, Audiovisual and Multimedia Systems, April 2013.

[4] B. Zatt, M. Shafique, F. Sampaio, L. Agostini, S. Bampi, and J. Henkel, "Run-time adaptive energy-aware motion and disparity estimation in multiview video coding" IEEE Design Automation Conference, pp. 1026-1031, 2011.

[5] C. Chen, C. Huang, Y. Chen, and L. Chen, "Level C+ data reuse scheme for motion estimation with corresponding coding orders", IEEE Transactions on Circuits and Systems for Video Technology, vol.16, no.4, pp. 553-558, 2006.

[6] M. Budagavi, and M. Zhou, "Video Coding Using Compressed Reference Frames", IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 1165-1168, 2008.

[7] D. Gupte, B. Amrutur, M. Mehendale, A. Rao, and M. Budagavi, "Memory Bandwidth and Power Reduction Using Lossy Reference Frame Compression in Video Encoding", IEEE Transactions on Circuits and Systems for Video Technology, vol. 21, pp.225-230, 2011.

[8] X. Bao, D. Zhou, and S. Goto, "A lossless frame recompression scheme for reducing DRAM power in video encoding", IEEE International Symposium on Circuits and Systems, pp.677-680, 2010.

[9] S. Lee, M. Chung, S. Park, and C. Kyung, "Lossless frame memory recompression for video codec preserving random accessibility of coding unit", IEEE Transactions on Consumer Electronics, vol.55, pp.2105-2113, 2009.

[10] T. Yng, B. Lee, and H. Yoo, "A low complexity and lossless frame memory compression for display devices", IEEE Transactions on Consumer Electronics, vol.54, pp.1453-1458, 2008.

[11] D. Silveira, M. Porto, and L. Agostini, "A Lossless Approach for External Memory Bandwidth Reduction in Video Coding Systems and Its VLSI Architecture", IEEE International Conference on Multimedia and Expo, 2013.