AN IMPROVED ALGORITHM FOR RATE DISTORTION OPTIMIZATION IN JPEG2000 AND ITS INTEGRATED CIRCUIT IMPLEMENTATION[1]

ZhiHua Wang, XiangXie, LiZhang and ChunZhang

Department of Electronic Engineering, Tsinghua University, P. R. Beijing China 100084

ABSTRACT

The Rate Distortion Optimization (RDO) plays an important role in JPEG2000 encoder. An improved RDO algorithm is presented in this paper. The proposed algorithm is suitable for integrated circuit implementation and can reduce the computational cost. A hardware architecture which includes control unit, memory, divider, data converter is also given to implement the algorithm. The circuit based on the improved algorithm has been integrated in a JPG2000 chip codec core.

1. INTRODUCTION

The block diagram of a JPEG2000 encoder /decoder is illustrated in Fig. 1.

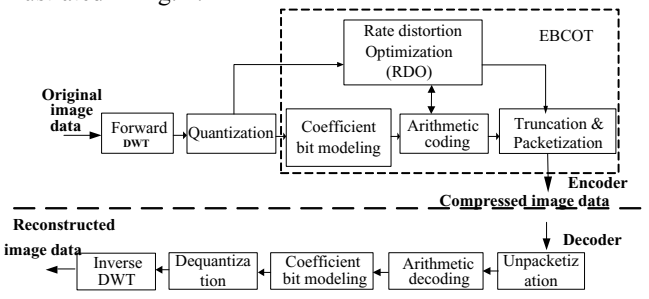

Fig. 1. Block diagram of the JPEG2000 encoder and decoder

In this figure, input of RDO from the arithmetic coding mainly include Weighted Mean Squared Error (WMSE), the number of code bytes of every truncation point and the number of truncation points in every block. The rate distortion slope which are output of RDO module are sent to the module of truncation and packetization to form the final code-stream. It is the rate distortion optimization, that enables the JPEG2000 to achieve a high compression performance. When the target rate distortion slope is known through the statistics of large number of images, arithmetic coding can compare the rate distortion slope from RDO with the target slope to reduce both the computational cost and the memory size of an encoder[1][2]. In this paper, an improved algorithm of RDO which is suitable for integrated circuit implementation is proposed. A comparison of proposed algorithm with other algorithm in VM8.0[3] is demonstrated to show the performance of the new algorithm.

This work is partly supported by National Natural Science Foundation of China 60372021

2. THE IMPROVED ALGORITHM

The process of the rate distortion optimization can be denoted by a constrained optimization system as follows:

$$Min \quad \sum_i D_i^k \qquad (1)$$

$$s.t \;\; R = \sum_i R_i^k <= R_{max} \qquad (2)$$

Where D_i^k is the Weighted Mean Squared Error (WMSE) of the truncation point k in the block i, and R_i^k is the number of code bytes of the truncation point k in the block i. From (1) and (2) equations, the Lagrange function can be expressed as the equation (3).

$$L = \sum_i (R_i^k + \lambda D_i^k), \; (R <= R_{max}) \qquad (3)$$

Where $\lambda^{-1} = \Delta D_i^k / \Delta R_i^k$ and λ may be interpreted as a quality parameter[3]. Therefore, an optimized set of truncation points can be obtained by the solution of equation (3).

The rate distortion optimization algorithm provided in VM8.0 [3] is described briefly in the following:

1) Set $N_i = \{ n \}$, i.e. the set of all candidate truncation points.

2) Set $p = 0$

3) For $k = 1, 2, 3, 4$,...

If k belongs to N_i

Set $\Delta R_i^k = R_i^k - R_i^p$ *and* $\Delta D_i^k = D_i^p - D_i^k$

Set $S_i^k = \Delta D_i^k / \Delta R_i^k$

If $p \neq 0$ and $S_i^k > S_i^p$ then remove p from N_i and go to step *2)*

Otherwise, set $p = k$.

For the convenience of description, the sequence of step *2)* -> step *3)* -> step *2)* is defined as an external cycle. The iteration inside step *3)* is defined as an internal cycle. It is obvious that the computational cost is spent mainly in the iteration between step *2)* to step *3)*. Inside step *3)*, the primary operations are division, subtraction and comparison. In the above algorithm, p is always set zero in step *2)*, which costs more clocks to implement those operations in step *3)*. In order to explain it clearly, an example is given in the following.

It is assumed that N_i is initialized by *{0,1,2,3,4,5,6,7}* and after several internal cycles, $p = 5$, $k = 6$ and $S_i^6 > S_i^5$ in step *3)*. Before jumping from the internal cycle to the step *2)*, the truncation point *5* should be removed from the candidate truncation points, and so $N_i = \{0,1,2,3,4,6,7\}$. Then, the value of p will be set zero again in step *2)*, which results in S_i^k *(k =1,2,3,4)* being computed again in step *3)* before computing S_i^6.

Such computational cost is not necessary because the values of S_i^k *(k = 1,2,3,4)* have been computed in the previous external cycles. In fact, only $S_i^6 = \Delta D_i^6/\Delta R_i^6$ need be computed in this internal cycle. Here $\Delta R_i^6 = R_i^6 - R_i^4$ and $\Delta D_i^6 = D_i^4 - D_i^6$ because the truncation point *5* doesn't belong to the candidate truncation point set, N_i. Even if in this internal cycle, there is no computational cost before computing S_i^6 in the special case when those candidate truncation points *1,2,3* and *4* all have been removed from the set N_i in the previous external cycles, computing S_i^6 still costs five clocks to decide whether those truncation points *1,2,3* and *4* belong to the set N_i or not for p is always set zero in step *2)*.

Above process of RDO can also be illustrated in Fig.2. Based on the information theory, the rate distortion curve should be monotonically decreasing and concave[4]. The process of RDO illustration is equivalent to link all the candidate truncation points except those convex points to form the concave curve. RDO illustration consists of two basic steps. The first step is to link a new candidate truncation point, which corresponds to the division and subtraction operations of the above algorithm. The second step is to decide whether this candidate truncation point is a convex point or not, and if the point is a convex point, it must be removed from the curve, which corresponds to the comparison and remove operation of the above algorithm. In Fig.2, candidate truncation point *0,1,2,3* and *4* are linked into the curve by turns because they are not convex points. After candidate truncation point *5* is linked and compared, the truncation point *5* must be removed from the curve because it is a convex point. Then, step *2)* must be executed and p is set zero based on the above algorithm. So, candidate truncation point *0,1,2,3* and *4* have to be linked by turns again before candidate truncation 6 is linked. Obviously, those linkage operations are not necessary. This is caused by the fact that p is always set zero in step *2)*.

Based on the analysis above, an improved algorithm is suitable for hardware implementation can be described as follows:

1) Set $N_i = \{ n \}$, i.e. the set of all candidate truncation points.

2) Set $p = q = 0$,

3) For $k = q+1, q+2, q+3, q+4, ...$

If k belongs to N_i

Set $\Delta R_i^k = R_i^k - R_i^p$ and $\Delta D_i^k = D_i^p - D_i^k$

Set $S_i^k = \Delta D_i^k/\Delta R_i^k$

If $p \neq 0$ and $S_i^k > S_i^p$ then remove p from N_i, set $q = k-1$ and go to step *4)*

Otherwise, set $p = k$.

4) For $k= p-1, \ldots, 4, 3, 2, 1, 0$

If k belongs to N_i

Set $p = k$ and go to step *3)*

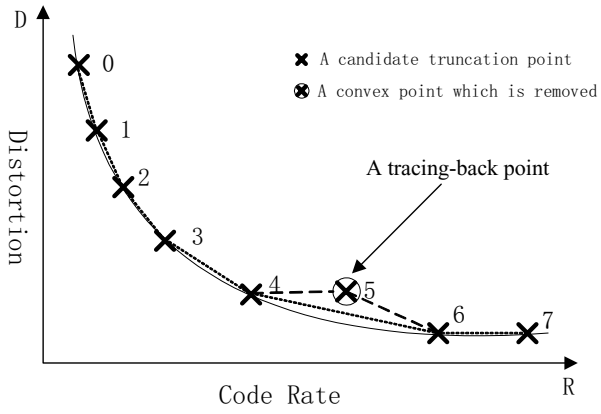

Fig.2. Rate distortion optimization

Still using the previous example, for $S_i^6 > S_i^5$, only truncation point *5* is removed from the set of the candidate truncation points and then $N_i = \{0,1,2,3,4,6,7\}$. Before discussing it, three definitions are given as follows:

1) The recurrence, step*3)* -> step*4)* -> step*3)*, is defined as '*a tracing-back operation*'.

2) The candidate truncation point like truncation point *5* is defined as '*a tracing-back point*'.

3) The distance between the first candidate truncation point zero and the tracing-back point in a tracing-back operation is defined as '*maximum tracing-back depth*'.

Table. I. the number of tracing-back operation in three pictures

Name	Size	Bit depth	Total pass number	Tracing-back Number	Total Maximum Tracing-back depth
Lena	512*512	8	3985	862	5819
Baboon	512*512	8	4801	799	5266
Plane	512*512	8	4033	901	6907

In this example, the tracing-back point is the candidate truncation point 5 and maximum tracing-back depth equals *5* (i.e. $5 - 0 = 5$) in the tracing-back operation when $S_i^6 > S_i^5$ happens. In this improved algorithm, before computing S_i^6, only one clock is spent in step *4)* and S_i^k $(k = 1,2,3,4)$ needn't be computed. In comparison with the original algorithm, the conclusion can be drawn that the larger total tracing-back operation number and total maximum tracing-back depth there are in an image, the more computational cost and clocks will be saved in this improved algorithm. Based 2-D discrete wavelet transform (DWT) with the 9/7 filter, the number of tracing-back operations and maximum tracing-back depth in three typical images are got and shown in table.I. Total pass number in this table means total candidate truncation point number in an image. Total tracing-back number and total maximum tracing-back depth are both relatively large to the total pass number. So, the improved algorithm can reduce the computational cost and clocks significantly.

3. HARDWARE IMPLEMENTATION

a) Architecture

The architecture of an implementation of RDO algorithm is shown in Fig.3. The RDO module consists of control unit including a system controller and a computation controller, a data converter, a dual port block memory and a divider. The input data from the arithmetic coding mainly include Mean Squared Error (MSE), scale weight factors of MSE, the number of code bytes of every truncation point and the number of truncation points in a block. The length of those data are 32 bits, 8 bits, 6 bits and 11 bits respectively based on the assumption that the block size is 32*32 and the implementation precision of wavelet coefficient is 16 bits in JPEG2000 coding. The final output result is rate distortion slope with 16 bit data width. Every part of this architecture is discussed respectively in detail in the following.

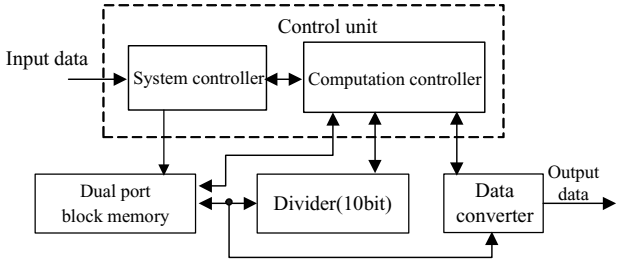

Fig. 3. Hardware architecture of RDO algorithm

b) Dual Port Block Memory

Since the data stored in memory need to be read out and written in simultaneously, a dual port block memory is required. The organization of the data is shown in Fig.4. Cumulative_bytes stands for the number of code bytes of every truncation point, *abs(m-n)<=1*, and *min{m,n-1}=< k <= min{m,n}* in this figure. Considering that the RDO process of one block is not finished untill arithmetic coding of this block is finished, a ping-pong structure is introduced in the memory. This ping-pong structure assures that arithmetic coding can continuously output coded bytes to the module of RDO even if computation of rate distortion slope of the previous block hasn't finished yet. Considering that the first bitplane is scanned only in one pass, namely, cleanup pass, the maximum number of passes in a block can be got from the equation (4).

$$MaxPassNumber=(precision - signbit - extra\_lsb - 1)*3+1 \quad (4)$$

Where precision is the precision of wavelet coefficient and extra_lsb is quantization step size. It is assumed that the size of the image to be compressed is 512*512 and the block size is 32*32 in JPEG2000 coding. The precision of the LL sub-band is 13bits and is higher than any other sub-bands. So, the possible maximum number of passes is *(16-1-2-1)*3+1=37* and the total depth of the memory is 37*3=111. The width of the memory is 43bits.

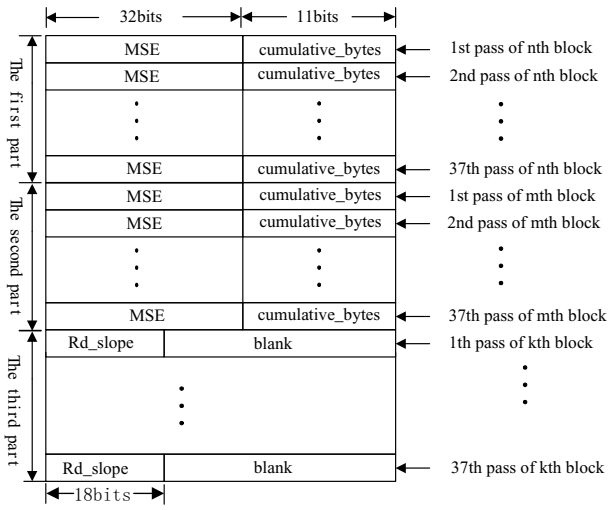

Fig.4. Assignment of the memory.

c) Control Unit

The control unit is the core of RDO and composed of two parts. One is a system controller to implement the control of acquisition and storage of input data. The other is a computation controller to implement the optimized algorithm of RDO. The basic operations of the computation control unit can be summarized as follows.

1) Reading operation.

Read data from the dual port block memory including MSE, cumulative bytes and rate distortion slope.

2) Computing operation.

Output MSE, scale weight factor and cumulative bytes to the divider to compute the rate distortion slope .

3) Comparing operation.

Compare 18 bit rate distortion slope of current truncation point with that of previous truncation point.

4) Writing operation.

Write 32 bit MSE, 11 bit cumulative bytes and 18 bit computed rate distortion slope into memory.

5) Outputting operation.

Output 18 bit rate distortion slope to the data converter to get 16 bit rate distortion slope.

The tracing-back operation mainly includes reading operations, computing operations and comparing operations. A 37 bit register is required to indicate which truncation points have been removed from the set of candidate truncation points. Those removed truncation points needn't be considered any more, so lots of computational cost and time can be saved.

d) Data Converter

The final result of rate distortion slope is 16 bit float point number including 7 bit exponent and 9 bit mantissa, which is illustrated in Fig. 6.

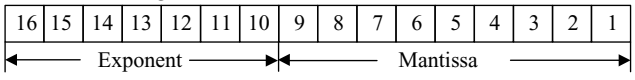

Fig.6. The format of the rate distortion slope

The data from dual port block memory, 18bit float point number, are converted into the final result by data converter.

e) Divider

The divider implements the function, $\Delta D_i^k/\Delta R_i^k$, and outputs an 18 bit float point number. The design of the divider mainly includes the following three steps:

1) Precision of the division.

For the mantissa of the final output result of rate distortion slope is *9* bits in Fig. *6*, scratchpad data and output data of the divider need only *10* bit mantissa to guarantee the final precision, 9 bit mantissa. Under the condition of this precision, the experimental results show that the PSNR (Peak Signal Noise Ratio) is not affected.

2) Data transformation.

ΔR_i^k is a 11 bit integer and ΔD_i^k is a *32* bit float point number. $\Delta D_i^k/\Delta R_i^k$ can be got from the equation (5). In this equation, ΔMSE is a *32* bit integer and scale weight factor can be expressed by the division of two integers[3], i.e. weight_factor1/weight_factor2. There are two division operations in this equation. In order to reduce one division operation, it can be transformed to the equation (6).

$\Delta D_i^k/\Delta R_i^k=(\Delta MSE*weight\_factor1/weight\_factor2)/\Delta R_i^k$ (5)

$\Delta D_i^k/\Delta R_i^k=(\Delta MSE*weight\_factor1)/(\Delta R_i^k*weight\_factor2)$ (6)

Considering that final result of rate distortion slope is a float point number, we must transform the dividend and the divisor from integer to 18 bit float point number of which format is illustrated in fig. 7.

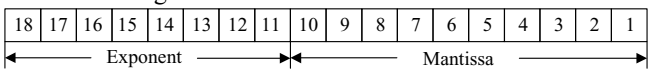

Fig.7. The format of dividend and divisor

The key of the transformation operation is to get the position of nonzero MSB (Most significant bit). Considering the tradeoff between time and area, this paper presents a novel algorithm of the transformation operation. The algorithm includes the following steps:

a) Divide the object data into several segments.

The length of each segment depends on the tradeoff between time and area. In our system, a 32 bit data is divided into four segments and the length of each segment is 8 bits.

b) Get the position of the first nonzero MSB of each segment.

Supposed that there are n bits in every segment, we can get the equation (7).

$$
\begin{aligned}
r_{k,n-1} &= b_{k,n-1} \\
r_{k,n-2} &= \bar{b}_{k,n-1} b_{k,n-2} \\
&\cdots \\
r_{k,i} &= \bar{b}_{k,n-1}\bar{b}_{k,n-2}\ldots\bar{b}_{k,i+1} b_{k,i} \qquad (7) \\
&\ldots \\
r_{k,1} &= \bar{b}_{k,n-1}\bar{b}_{k,n-2}\ldots\bar{b}_{k,i+1}\bar{b}_{k,i}\ldots\bar{b}_{k,2} b_{k,1} \\
r_{k,0} &= \bar{b}_{k,n-1}\bar{b}_{k,n-2}\ldots\bar{b}_{k,i+1}\bar{b}_{k,i}\ldots\bar{b}_{k,2}\bar{b}_{k,1} b_{k,0}
\end{aligned}
$$

Where $b_{k,n-1}b_{k,n-2}\ldots b_{k,i+1}b_{k,i}\ldots b_{k,1}b_{k,0}$ stands for the kth segment and $r_{k,n-1}r_{k,n-2}\ldots r_{k,1}r_{k,0}$ is output result which indicates the position of the first nonzero MSB of the kth segment.

c) Get exponent and mantissa.

After acquiring the position of the first nonzero MSB of each segment in step b), we can get the highest nonzero segment k, where $r_{k,n-1}r_{k,n-2}\ldots r_{k,1}r_{k,0} \neq 0$, and the global position of nonzero MSB. Then, exponent and mantissa can be got easily.

3) Design of divider

Through the analysis above, the dividend is 10 bits and divisor is 21 bits in the divider. The division can be expressed as follows:

$$\frac{1.a_8a_7\ldots a_1a_0\,0_{10}0_9\ldots 0_0}{1.b_8b_7\ldots b_1b_0}$$

In this system, the division algorithm is based on radix-4 division with operands scaling[5].

f) Performance

The post simulation of this circuit demonstrates that this circuit doesn't affect the image PSNR comparing with the algorithm in VM8.0. Given the 0.25μm CMOS technology of DONGBU, the total scale of this circuit is approximately 21,923 gates with a 9.76 ns critical path delay. The circuit has been integrated successfully in a JPG2000 chip codec core.

4. CONCLUSION

An improved algorithm that is suitable for integrated circuit implementation of RDO and the design of its circuit have been presented in this paper. Compared with the algorithm in VM8.0, the proposed algorithm can decrease a great deal of computational cost. The circuit based on the improved algorithm has been integrated in a JPG2000 chip codec core. In addition, the circuit of RDO and other circuits of the JPEG2000 encoder will be further simplified through the predictive truncation[1][2], which will be our next work.

5. REFERENCES

[1] Taubman, D. "Software architectures for jpeg2000" Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on , Volume: 1 , 2002, Page(s): 197 -200

[2] Masuzaki, T.; Tsutsui, H.; Izumi, T.; Onoye, T.; Nakamura, Y. "Adaptive rate control for JPEG2000 image coding in embedded systems" Image Processing. 2002. Proceedings. 2002 International Conference on , Volume: 3 , 2002. Page(s): 77 -80

[3] C. Chritopoulos (editor), "JPEG2000 Verification Model 8.0 (technical description)." ISO/IEC JTC1/SC29/WG1 N1822, July 21, 2000

[4] Berger,T "Rate Distortion Theory" Englewood cliffs, N.j. Prentice-Hall, 1971

[5] M.D. Ercegovac and T. Lang, "Simple Radix-4 Division with Operands Scaling," IEEE Trans. Computers, vol. 39, no. 9, pp. 1,204-1,207, Sept. 1990