2014 The 2nd International Conference on Information Technology and Science (ICITS 2014)
IPCSIT vol. 59 (2014) © (2014) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2014.V59.31

Picture-level Motion Information Optimization in Video Coding using Simulated Annealing

Gang Wu [1] and Houqiang Li [1 +]

[1] University of Science and Technology of China, Hefei, China

Abstract. In this paper, we present a picture-level motion information optimization method using simulated annealing to improve the efficiency of video coding. Based on the simulated annealing search algorithm, we randomly select a group of macroblocks out of a picture and then modify motion information of those macroblocks according to previously defined rules. Experimental results indicate that this method works effectively in terms of improving the encoding performance.

Keywords: video coding, simulated annealing, motion estimation, picture-level optimization

1. Introduction

In the past decades, the world saw an explosive increase in scenarios which digital video was widely used and devices were also developing rapidly. In the wake of development of processing speed, storage capacity, and screen resolution, the demand of watching videos of high resolution and quality is increasing. In order to compress videos into occupying less storage but retaining a good visual quality, several video coding standards have been proposed to make use of the redundancy in spatial, temporal and entropy domain. In 2003, Joint Video Team (JVT) developed the H.264/AVC (Advanced Video Coding) standard [1]. H.264/AVC is a block-based video coding standard and is much more effective than previous standards such as MPEG-1, MPEG-2, MPEG-4, H.263 because of its complicated design and innovatively added useful tools to improve subjective quality of the videos.

A useful tool in H.264/AVC is Motion Estimation (ME) [2]. In brief, in block-based video coding standards, a picture in divided into several blocks which share an identical size. For any picture using inter prediction, ME is performed for each block to search the most similar region from other encoded pictures. Nevertheless, there are some problems with ME. Firstly, the contents in videos are apparently different: most parts in video conferences and video surveillance are almost stationary; colour in the animated cartoons is simple and smooth but edges of those are sharp; details of movies are complex; sports broadcasting records fast moving objects; videos recorded by mobile phone users are of low quality, etc. Moreover, devices that are used to record and display have different resolutions but coding efficiency is partly dependent on the resolution. It is commonly known that the detail of one block of low resolution is more complicated than that of a higher resolution, the coding efficiency under these two conditions are definitely different. Furthermore, ME is far from perfect since it doesn't fully make use of the relativity of neighbouring blocks but only considers the motion information of left and upper blocks. However, blocks in the vicinity such as blocks in the right of or below the current block may also provide useful information.

In [3], the author proposed a method considering the distortion of successive pictures to reduce the flickers, but it only applies to intra-coded pictures. Authors of [4] proposed a model to re-allocate bits based on motion saliency in one picture to improve visual quality, however, it didn't exploit the motion

+ Corresponding author. Tel.: + 8613965122221; fax: +865513601342.
E-mail address: lihq@ustc.edu.cn.

information of neighbouring blocks. Reference [5] analysed the statistical characters of Motion Vector (MV) of a whole picture and proposed an early-termination approach in ME. Another approach proposed in [6] improved the performance of Enhanced Predictive Zonal Search (EPZS) search algorithm for ME and reduced the complexity. The authors of [7] proposed a hexagon-based zonal search in order to obtain a same visual quality compared with full search but with low computation complexity. Reference [8] proposed fast mode decision for inter prediction, which exploited the cost after ME of each possible mode according to a pre-defined rule for an early-termination determination. Nevertheless, these methods aim at reducing the complexity but the coding performance is not increased and the relativity of neighbouring blocks is not utilized yet such that there are still plenty of spaces to progress.

In order to tackle these problems, we propose a method to optimize the motion information in the picture-level after finishing standard processes of H.264/AVC. In our method, simulated annealing search algorithm [9] is harnessed to randomly select a set of blocks from the current picture, modify their motion information and accept some modifications according to our criterion. Experimental results show that our proposed method can obviously improve the coding efficiency, especially for video sequences that have fierce motion or have low resolutions.

2. Picture-Level Motion Information Optimization

2.1. Background of H.264/AVC

In H.264/AVC, every picture is divided into several square blocks, which is called marcoblock (MB) whose size is 16 pixels wide and 16 pixels high. In practical use, one picture is often separated into several slices, and each slice usually contains a number of successive MBs. Every two slices are not overlapped. Every MB will be encoded sequentially and independently according to its coding order, which means current MB cannot be processed until all MBs prior to the current MB have been encoded. The best encoding mode of each MB is determined through mode decision, which goes through every possible mode and then compares coding performance of each available modes to select the best. After that, best mode is used to perform prediction, transformation, quantization and entropy coding. After all these processes of one MB have been done, the next MB will repeat all these processes until the end.

Literally in the pictures encoded as P-Slice or B-Slice, ME is performed for each MB in them. For one MB, each possible mode will exploit ME to determine the best MVs and the best reference pictures. The mode achieving the best balance between distortion and bit-rate is regarded as the best mode for this MB.

The criterion of coding efficiency is Rate-Distortion Cost (RD-Cost) [10] which is effective to evaluate the performance of block-based video coding standards. And also it is exploited to choose the best encoding mode in H.264/AVC. The encoding modes of each MB are derived without considering the relativity of neighboring MBs because this criterion is performed independently when encoding one MB.

2.2. Proposed Method

At the very beginning, we define the target function similar to the RD-Cost model

$$e(i) = D(i) + \lambda \cdot R(i)$$

where $D(i)$ denotes distortion between the original picture and the reconstructed picture at state i and $R(i)$ denotes the bit-rate of the same picture. Parameter λ, which is determined uniquely by the Quantization Parameters (QP), is derived in the same way in H.264/AVC.

As for the distortion $D(i)$, the Sum of Squared Errors (SSE) is used

$$D(i) = \sum_{j \in P} [org_j - rec_j(i)]^2$$

where P denotes the set of all pixels in one picture and j denotes any pixel. Variables org_j and rec_j represent pixel value of j in the original picture and reconstructed picture respectively. $R(i)$ is the sum of all bit-rate used to encode current picture.

After all MBs in a P-Slice or B-Slice have been processed, we regard it as the initial state i_0.

The pseudo code in Tab. 1 shows our proposed method *MotionInfo_SimuAnl*() briefly. Note that only for pictures which are encoded as P-Slice or B-Slice, this procedure will be executed and each of these picture call this procedure independently.

Firstly, calculate distortion $D(i_0)$ and bit-rate $R(i_0)$ of current picture and then use (1) to compute the current target function $e(i_0)$. After that, technically the initial state is stored as the best state for future comparison, though it is not the optimal state in common cases.

Secondly, in function *modify_mv*(), we select N MBs at random from the current picture and modify their MVs slightly and randomly, stepping in to a new state i_{new}. Afterwards, calculate the current target function $e(i_{new})$. And then compute the probability of state i_{new}. For simplification, the probability calculation in our method is not Boltzmann distribution in [9] but is written as

$$P \mid E = e(i) \mid = C_1 \cdot \exp[-e(i) / C_2]$$

where C_1 and C_2 are constants. Note that the number N should be assigned to a large number as simultaneously changing MVs of a large number of MBs would probably lead to unsatisfactory, even worse performance. It is more intelligent to adaptively set the number N according to the value of target function.

Further, obtain a random number using the function *random*(), of which the range is 0 to 1, inclusively. Compare the random number we got with $P|E=e(i_{new})|$. Accept the new state i_{new} when $P|E=e(i_{new})|$ is larger than that random number and use i_{new} as the starting point in the next round; otherwise, use the previous state i to repeat the while loop in Table. 1.

In the next step, compare the target function of current state and that of the best state. If that of the current state i_{new} is less than the best state, save the data of current state as the best state. Apparently, we always store the best state so that when the termination condition is reached, we can restore the best state as its ultimate state since it is the optimal result we can ever get during this procedure.

Table. 1: Pseudo Code of Proposed Method.

```
1:   procedure MotionInfo_SimuAnl()
2:     i = i0; e = e(i)
3:     ibest = i; ebest = e                    // store initial state
4:     k = 0
5:     while k < kmax and e > emax             // terminating condition
6:       inew = modify_mv ()                   // randomly modify MV
7:       enew = e(inew)
8:       if P|E=e(inew)| > random() then
9:         i = inew; e = enew
10:      if enew < ebest then                  // store best state
11:        ibest = inew; ebest = enew
12:      k = k+1
13:    return ibest
14:  end procedure
```

It is necessary to explain the key function *modify_mv*()in line 6 in the Tab. 1. After all the MBs in a picture have been processed, every MB has been determined the best mode, including the partition pattern, the best reference pictures, etc. In this function, as we have mentioned before, we randomly choose N MBs, for instance N equates to 2, and modify their motion information slightly. When modifying one MV, we need to randomly select any one component of its vertical component or a horizontal component, and add the very component to a small integer, in most cases ranging from -4 to 4 in our implementation. Note that we only modify the MVs in the MB but all other information such as *mb_type*, *sub_mb_type*, *transform_8x8_flag* still remain unaltered. For example, if one of the MBs in a P-Slice is chosen by our algorithm and it consists of two 16x8 divisions. We may just randomly choose only one 16x8 division that has only one MV each or both two divisions to modify those MVs. So after all N MBs have been randomly selected and modified, it is time to perform processes such as inter prediction, motion compensation, transformation, quantization, de-blocking filtering and entropy coding for every MB. Then calculate Peak Signal-to-Noise Ratio (PSNR) and bit-rate of this picture to acquire statistics.

In order to obtain an equilibrium between the coding efficiency and subjective quality of encoded pictures, it is vital to set an appropriate termination condition. In our design, after this proposed method has been executed more than k_{max} times or the minimum target function of the current state is not greater than

e_{max}, this algorithm will be terminated; otherwise it will keep searching for a better state. Note that the value k_{max} can vary according to the video coding environment such as the speed of the computers and the number of threads used. Likewise e_{max} can also be changed as k_{max}.

In a nutshell, this algorithm is not complicated. It can be readily implemented in widely-used programming languages or in popular platforms. Moreover, it can be easily modified to run parallel utilizing multiple threads when available such that it can perform faster and more efficiently.

3. Experiment

3.1. Test Conditions

The experiments are carried out based on the H.264/AVC reference software JM 18.4 [11]. Most tools are set default in high profile. To be specific, we enable de-blocking filter, RD optimization of high complexity. Also, we disable bi-directional prediction for block partitions whose sizes are less than 8x8, weighted prediction, and rate control. Every picture contains only one slice. Hierarchical coding is used and the size of Group of Pictures (GOP) is set to 8. The number of reference pictures is set to 1. Intra period is set to 32. EPZS search method is used for ME. In inter pictures the intra prediction mode is disabled. The resolutions of test sequences are 176x144 (QCIF), 352x288 (CIF), 416x240, 832x480, 1280x720, and 1920x1080. For every sequence, 100 pictures are encoded and tested, and QPs are set to 24, 28, 32, and 36 respectively. BD-Rate [12] is exploited to evaluate the performance of our method.

3.2. Experimental Result

Table. 2: Performance of Sequences of Different Resolutions.

Sequence	QP	H.264/AVC Standard				Proposed Method				BD-Rate Improvement		
		Bit-rate (kbps)	PSNR-Y (dB)	PSNR-U (dB)	PSNR-V (dB)	Bit-rate (kbps)	PSNR-Y (dB)	PSNR-U (dB)	PSNR-V (dB)	Y	U	V
Crew 176x144@30Hz	24	332.98	39.90	43.68	42.26	318.96	39.93	43.83	42.40	4.72%	6.06%	6.27%
	28	176.94	36.93	41.61	39.81	172.55	37.01	41.71	39.91			
	32	95.15	34.13	39.80	37.68	92.84	34.26	39.90	37.86			
	36	49.97	31.57	37.99	35.71	49.25	31.71	38.10	35.84			
Foreman 352x288@30Hz	24	944.71	39.22	43.47	46.37	930.84	39.29	43.52	46.45	3.41%	3.06%	4.07%
	28	525.42	36.77	41.71	44.46	518.12	36.83	41.75	44.54			
	32	306.52	34.34	40.50	42.79	300.28	34.42	40.53	42.88			
	36	185.67	31.97	39.33	41.33	184.09	32.14	39.38	41.40			
Basketballpass 416x240@50Hz	24	1484.51	40.31	43.30	42.99	1462.27	40.37	43.35	43.09	2.30%	3.31%	4.39%
	28	850.98	37.23	41.01	40.46	841.88	37.29	41.10	40.57			
	32	478.11	34.37	39.23	38.46	472.15	34.42	39.30	38.60			
	36	270.04	31.91	37.67	36.36	266.45	31.97	37.70	36.49			
Basketballdrill 832x480@50Hz	24	6942.00	38.98	42.15	42.45	6866.65	38.99	42.16	42.46	1.50%	1.68%	1.90%
	28	3837.33	36.40	40.01	39.95	3795.08	36.41	40.03	39.98			
	32	2195.12	34.04	38.13	37.86	2172.62	34.06	38.16	37.89			
	36	1288.00	31.86	36.37	35.82	1265.62	31.87	36.37	35.85			
Kristenandsara 1280x720@60Hz	24	4204.77	42.34	46.64	47.58	4196.19	42.36	46.65	47.58	1.23%	0.43%	0.28%
	28	2227.03	40.59	45.00	45.82	2229.92	40.63	45.01	45.83			
	32	1325.32	38.58	43.52	44.25	1322.27	38.64	43.53	44.26			
	36	820.73	36.40	41.83	42.45	824.26	36.46	41.83	42.45			
Kimono 1920x1080@24Hz	24	12161.98	41.08	43.14	44.93	12062.41	41.07	43.14	44.94	1.20%	2.27%	2.07%
	28	7063.85	39.16	41.82	43.35	6983.51	39.11	41.83	43.36			
	32	4062.56	36.90	40.66	42.16	3974.02	36.92	40.70	42.22			
	36	2291.72	34.35	39.58	41.14	2287.09	34.37	39.58	41.12			

In Table. 2, we list one sequence for each resolution. It is obvious that, in most cases, our proposed method can result in an increase of PSNR in all Y, U and V components as well as a conspicuous decline in bit-rate. For instance, the sequence “Crew” whose resolution is QCIF reaches the largest improvement in BD-Rate whereas sequence “Kimono”, of which the resolution is 1920x1080, achieves the lowest improvement of BD-Rate of Y component.

Some exceptions do exist that when PSNR is increased but the bit-rate is also increased. For example in Tab. 2, the bit-rate of sequence “Kristenandsara” is increased using our method. Since the target function is the sum of distortion and bit-rate, if the extent of decrease in distortion (increase in PSNR) is much more than that of the increase in bit-rate, the performance is still improved. Likewise, if the extent of decrease in bit-rate is much more than that of decrease in PSNR, the performance is also improved.

It is clear that the four sequences in Tab. 4 obtain different performance. Sequence “Crew” is about a group of astronauts moving towards the camera in the meanwhile the camera is moving from left to right, therefore motion between consecutive pictures is unsmooth. However, the motion in sequence “Mobile” is relatively smooth because it records the scene that a model train is moving from right to left slowly and the background is moving at a constant speed from right to left. Thus, we can conclude that the original motion information of H.264/AVC, is relatively precise when the motion is not intense. When motion is unsmooth, our proposed method can obtain apparent improvement.

Table. 3: Average BD-Rate Improvement

Resolution	BD-Rate Improvement		
	Y	U	V
176x144	3.16%	6.88%	4.40%
352x288	2.31%	5.49%	3.00%
416x240	1.79%	2.33%	2.93%
832x480	1.39%	1.81%	1.98%
1280x720	1.02%	0.42%	0.39%
1920x1080	0.56%	0.69%	0.72%

Table. 4: Performance of Qcif Sequences

Sequence	BD-Rate Improvement		
	Y	U	V
Mobile	1.42%	2.24%	2.21%
Highway	2.70%	15.24%	6.09%
Crew	4.72%	6.06%	6.27%
Soccer	3.80%	3.98%	3.01%
Average	3.16%	6.88%	4.40%

Look back to Tab. 2, it is easy to find out that the higher the resolution, the less improvement of BD-Rate it can obtain. Seen from Table III which records average improvements of all sequences tested for every resolution respectively, the percentage of improvement of Y component is decreasing as the resolution is increasing. The explanation is relevant to the proportion of the RD-Cost of an individual MB accounting for the entire RD-Cost of the picture. For instance, QCIF sequences contain only 99 MBs and every modification of MB would probably cause an obvious change in the RD-Cost of the entire picture. Nevertheless, a sequence whose resolution is 1280x720 comprises 3600 MBs which means that a small change in a single MB may slightly affect the whole RD-Cost of the picture at most. Further, in any picture of a sequence of low resolution, the video content in a MB may contain complex textile, for example the MB contains 1/4 of the face of someone in the video so that it need at least 4 MBs to include the whole face. But if the same visual content is presented in a higher resolution, a MB may only contain one tenth of the same face so that it is much smoother. When the details of MBs are complicated, the coding performance will fall as it is block-based. Therefore motion information of sequences of low resolution is more imprecise than that of high resolution.

3.3. Analysis

Generally, our proposed method is effective to improve the encoding performance of pictures using inter prediction. We observed that most modifications of MVs are very slight, for example a MV may change from (4, 2) to (4, 0) or from (5, -11) to (6, -10). Only a small portion of changes is relatively high. This indicates that the result of ME procedure in H.264/AVC is accurate to some extent but is not optimal. Our proposed method plays the role of refining the result of ME. Moreover, the improvement of different pictures encoded is also different according to corresponding content and motion of those pictures. If two consecutive pictures are almost stationary and most parts of them are identical, or the majority of regions in the picture are smooth, the improvement is relatively low, and vice versa.

In H.264/AVC, MBs of a picture are relatively independent due to the reality that every MB only has to consider its own RD-Cost when performing encoding process. In this fashion, it neglects the possibility that neighbouring MBs including this MB may represent an integral region such as a house or a mountain. Although in H.264/AVC standard, some processes utilize the information of left MB and upper MB in order to save bit-rate such as the process for obtaining MVP (Motion Vector Predictor) need to predict current MV according to left and upper MBs, the effectiveness of usage of those neighbouring MBs is limited. As mentioned before, all MBs in the same picture has to be encoded in coding order. Modifying a specific MB means all MBs after this MB in coding order will be affected since they rely on the modified MB to perform subsequent prediction and entropy coding. Our proposed method makes use of the simulated annealing

algorithm to find out MVs that are more appropriate for the entire picture. Using this proposed method to replace a new MV of an existing one in a MB, the RD-Cost of that MB may decrease whereas the RD-Cost of the entire picture will be increased. Hence the proposed approach can elevate the effectiveness via utilization of the relativity of the whole picture.

4. Conclusion

In this paper, we propose a method for H.264/AVC, to improve the coding efficiency. In our proposed method, we randomly choose a group of macroblocks in one picture and modify their motion vectors based on simulated annealing. Experimental results indicate that this proposed method can effectively obtain an obvious improvement in coding performance; therefore, it can effectively be applied to many offline application scenarios such as movies being played in the cinema, videos stored in the websites or DVDs, recorded videos from users, and so forth.

5. References

[1] *Advanced Video Coding for Generic Audiovisual Services*, ITU-T Rec. H.264 and ISO/IEC 14496-10 (MPEG-4 AVC), ITU-T and ISO/IEC JTC 1.

[2] P. Y. S. Cheung, "Motion Estimation in Video Coding," *in Proc. IEEE Region 10 Annual Conference on Speech and Image Technologies for Computing and Telecommunica-tions*, Brisbane, Qld., Australia, vol.2, pp.707, Dec. 1997. doi: 10.1109/TENCON.1997.648519.

[3] S. S. Chun, J. R. Kim, and S. Sanghoon, "Intra Prediction Mode Selection for Flicker Reduction in H.264/AVC," *IEEE Trans. Consumer Electron.*, vol.52, no.4, pp.1303-1310, Nov. 2006. doi: 10.1109/TCE.2006.273149.

[4] Z. Liu, H. Yan, L. Shen, Y. Wang, and Z. Zhang, "A Motion Attention Model Based Rate Control Algorithm for H.264/AVC," *in Proc. IEEE/ACIS International Conference on Computer and Information Science*, Shanghai, China, pp.568-573, June 2009. doi: 10.1109/ICIS.2009.165.

[5] J. Xu, Z. Chen, and Y. He, "Efficient Fast ME Predictions and Early-Termination Strategy Based on H.264 Statistical Characters," *in Proc. Pacific Rim Conference on Information, Communications and Signal Processing*, Singapore, vol. 1, pp. 218-222, Dec. 2003. doi: 10.1109/ICICS.2003.1292446.

[6] H. Y. C. Tourapis and A. M. Tourapis, "Fast Motion Estimation within the H.264 Codec," *in Proc. International Conference on Multimedia and Expo*, Baltimore, Maryland, vol. 3, pp. 517, July 2003.

[7] P. De Pascalis, L. Pezzoni, G. Mian, and D. Bagni, "Fast motion estimation with size based predictors selection hexagon search in H.264/AVC encoding", *in Proc. European Signal Processing Conference*, Vienna, Austria, Sep. 2004. doi: 10.1109/ICME.2003.1221362.

[8] Z. Zhou and M. T. Sun, "Fast Macroblock Inter Mode Decision and Motion Estimation for H.264/MPEG-4 AVC," *in Proc. International Conference on Image Processing*, Singapore, vol. 2, pp. 789-792, Oct. 2004. doi: 10.1109/ICIP.2004.1419416.

[9] S. Kirkpatrick, C. Gelatt, and M. Vechhi, “Optimization by Simulated Annealing,” *Science*, vol. 220, no. 4598, pp. 671-680, May. 1983.

[10] M. Karczewicz and Y. Ye, "Rate Distortion Optimized Quantization," *Document VCEG-AH21*, Antalya, Turkey, Jan. 2008.

[11] Joint Video Team of ISO/IEC MPEG and ITU-T VCEG, *H.264/AVC, Reference Software JM18.4*.

[12] G. Bjontegaard, "Calculation of Average PSNR Differences between RD Curves," *Document VCEG-M33*, Austin, TX, USA, Apr. 2001.