Hardware Architecture for Motion Estimation on Chrominance Samples

Gustavo Wrege, Ruhan Conceição, Marcelo Porto, Luciano Agostini

Group of Architectures and Integrated Circuits - GACI
Federal University of Pelotas - UFPel
Pelotas, Brazil
{gwgoncalves, radconceicao, porto, agostini}@inf.ufpel.edu.br

Abstract— **With the popularization of high definition digital videos the video coding process becomes even more important, due to the huge amount of bits needed to represent this kind of video. Nowadays, the video coding is mandatory in order to enable storing and broadcast of digital videos. ME is usually applied only at luminance information, the luminance motion vectors are simply reused by the chrominance samples. Thus, the chrominance information is not processed by the ME. This work shows a new approach, performing the ME process also at chrominance samples, in order to increase the image quality in high definition encoded videos, using fast algorithm in each video channel (Y, Cb and Cr). The simulation results for 10 HD 1080p videos, using DS, MDPS and S&IS algorithms shows an average PSNR gain of 2.385 dB in Cb and 3.022 dB in Cr channels. Moreover, it was proposed a hardware implementation for this approach which performs the ME with the MPDS algorithm, focusing on real time processing for high definition videos.**

Keywords— **Motion Estimation, Video coding, Chrominance samples.**

I. INTRODUCTION

Nowadays, it is notable the growth of devices which are able to perform digital videos. These devices require High Definition (HD) videos, which impresses the costumer with their image quality, mainly when compared to lowest definition videos. On the other hand, the increasing in the video resolution results in a significant increasing in the amount of data bits to be processed as well. Thus, video coding has been increasingly important at this scenario, since tasks such as: storing, broadcast and execution in not encoded HD videos are very costly.

Video coding uses techniques that exploit data redundancies, for instance: spatial, temporal and entropic. This work focus is the Motion Estimation (ME) process, which explores the temporal redundancies. This kind of redundancy is found on neighbor frames, and due to the high number of frames processed by second (24 to 30 for real time processing), this kind of redundancy is extremely high. Among the compression steps, the ME represents more than 80% of the process computational cost of current video encoders [1]. Despite its computational cost, ME is the main responsible for the gains in terms of data compression generated by the video coding process.

The emerging video coding standard HEVC (High Efficiency Video Coding) does not restrict the way how ME must be performed. The same occurs to the current video coding standard: H.264/AVC [2]. Thereby, this subject provides a large field of research in order to find new solutions considering mainly a relation between computation complexity and image quality. Moreover, the purposed solutions also should consider the hardware implementation, since software solution, for current technology, hardly reaches real time processing for high definition videos.

Usually, the video coding standard use the YCbCr color space. The component Y refers to luminance information and Cb and Cr refers to blue and red chrominance information respectively. In a video with color subsample of 4:2:0, approximately 66% of video data refers to luminance information. Hence, motion estimation is usually performed only on luminance samples. However, considering the increasingly dissemination of HD videos, motion estimation in chrominance blocks becomes necessary, since it could provide a significant increase on video quality.

The goal of this work is to propose a new approach for Motion Estimation, using chrominance information. Therefore, it is possible to increase visual quality, especially for HD videos. Moreover, this work proposes a hardware design for the ME on chrominance samples. Its design is based on a previous developed architecture for luminance samples as is presented in [5], which has excellent results in terms of quality and performance for luminance samples.

This paper is organized at this way: section 2 explains the motion estimation process. In Section 3 it is presented the software evaluation of motion estimation on chrominance samples. A hardware architecture for motion estimation on chrominance samples is proposed in section 4, and finally in section 5, it is present the conclusions about this work and future works.

II. MOTION ESTIMATION

Digital videos are composed by a sequence of images (frames) which are shown with some frequency. In order to generate movement sensation, it is necessary that about 30 frames be present per second [3]. The frames are divided into blocks and the blocks are divided in pixels, which are the minor video information. Pixels – in videos – represents bright information or color intensity.

Table I.

Results of quality and computational cost for DS, MPDS and S&IS algorithms

Videos	DS				DS Y/Cb/Cr				MPDS				MPDS Y/Cb/Cr				S&IS				S&IS Y/Cb/Cr			
	PSNR (dB)			CBC x10^8	Δ PSNR (dB)			Δ CBC x10^8	PSNR (dB)			CBC x10^8	Δ PSNR (dB)			Δ CBC x10^8	PSNR (dB)			CBC x10^8	Δ PSNR (dB)			Δ CBC x10^8
	Y	CB	CR		Y	CB	CR		Y	CB	CR		Y	CB	CR		Y	CB	CR		Y	CB	CR	
bluesky	30,82	36,94	39,79	0,22	-	1,16	0,73	0,35	34,17	38,53	41,27	1,23	-	0,42	0,16	1,93	25,94	32,80	36,73	0,98	-	1,60	0,89	1,70
man in ar	37,72	41,74	46,28	0,18	-	1,79	2,04	0,33	39,08	41,90	46,57	1,13	-	2,87	3,29	1,81	37,71	41,68	45,55	0,92	-	3,12	3,94	1,66
pedestrian area	32,20	41,53	41,11	0,25	-	2,68	3,96	0,35	34,74	42,19	41,81	1,69	-	3,38	5,30	2,07	32,38	39,21	38,86	1,06	-	3,55	5,17	1,72
riverbed	24,47	36,46	40,30	0,29	-	2,67	1,93	0,38	26,32	36,43	40,34	1,71	-	4,32	3,11	2,10	27,09	35,40	39,63	1,11	-	5,63	3,93	1,76
rolling tomatoes	37,02	40,73	42,34	0,19	-	2,06	4,94	0,34	37,71	40,79	42,62	1,36	-	3,15	6,45	2,11	36,47	37,64	38,51	0,97	-	6,03	9,86	1,75
rush hour	36,41	43,99	45,00	0,19	-	1,40	2,21	0,31	37,09	44,04	44,74	1,55	-	1,88	3,03	2,08	35,14	42,14	39,56	1,00	-	2,15	4,38	1,69
station2	38,04	43,67	43,84	0,20	-	1,13	1,20	0,31	38,57	43,74	43,94	1,33	-	1,49	1,45	1,91	32,89	40,72	40,33	0,97	-	3,21	2,26	1,67
sunflower	36,97	41,03	43,02	0,27	-	1,61	1,51	0,37	38,63	41,99	43,92	1,86	-	1,62	1,50	2,64	34,77	38,75	40,07	1,13	-	-0,18	1,13	1,83
tractor	29,79	36,23	37,38	0,30	-	1,21	1,75	0,40	31,86	37,37	38,22	1,72	-	1,13	1,89	2,33	29,24	35,26	32,73	1,14	-	1,40	4,06	1,83
traffic	26,09	34,95	37,25	0,32	-	2,59	2,52	0,40	29,75	36,70	39,43	1,80	-	3,14	3,13	2,28	28,66	34,80	37,43	1,16	-	3,36	2,95	1,81
Average	**32,95**	**39,73**	**41,63**	**0,24**	-	**1,83**	**2,28**	**0,35**	**34,79**	**40,37**	**42,29**	**1,54**	-	**2,34**	**2,93**	**2,13**	**32,03**	**37,84**	**38,94**	**1,04**	-	**2,98**	**3,86**	**1,74**

There is a lot of similarity among the frames which composes digital videos. This similarity is known as temporal redundancy [4]. ME aims to evaluate the temporal redundancies and reduces the amount of bits necessary to represents the video. For this purpose, ME uses a search area in an already processed frame (reference frame), comparing each block of the current frame with candidate blocks of the search area. Finally, it is found the most similar candidate block and the redundant information is mapped to motion vectors [5]. These vectors indicate the spacial displacement of the current block compared to reference frame. The similarity criteria used at this work was the Sum of Absolute Differences (SAD) [6].

Search algorithms were used in order to find the most similar block. These algorithms define how the search must be done inside the search area. The algorithm Full Search (FS) [6] presents the best results, however, it presents the highest computational cost . It occurs due this method that compares all the candidate blocks of the search area. In the literature there are several fast algorithms which aim to reduce the computational cost. These algorithms use some heuristics to speed up the search, decreasing the number of compared candidate blocks between current and reference frames. However, it results in quality losses at the video coding. Among the fast algorithms present at the literature, is possible to mention the Diamond Search (DS) [7], Multi Point Diamond Search (MPDS) [8], which is a hardware-friendly algorithm focused on HD videos, and also the Spread and Iterative Search (S&IS) [9] which uses a random strategy to achieve better quality on HD videos.

The results obtained by ME are evaluated through PSNR (Peak Signal-to-Noise Ratio) measured in decibels (dB), in a logarithm scale. The quantity of calculated candidate blocks (CBC) is used as performance metrics.

III. SOFTWARE EVALUATION OF MOTION ESTIMATION ON CHROMINANCE INFORMATION

Motion Estimation is usually performed only in luminance samples. However, these data represent just 66% of the video information. Thus, the remaining 34% of chrominance information are not processed by ME process. In the current video coding standard, the motion vector generated by the ME for the luminance samples is reused for the chrominance samples. Thereby, the results in terms of video quality for Cb and Cr channels are directly linked to these vectors.

This work focuses on to perform the ME process also on chrominance channels, in order to increase the image quality on high definition videos. For this purpose, three algorithms available in the literature were evaluated: DS, MPDS and S&IS, where these two last algorithms were previously developed by our research group. These algorithms were chosen due their results in terms of image quality and computational cost when applied for HD videos.

The software evaluation was developed as follows: firstly, it was generated results for the three algorithms described above, only for the luminance samples, and the resultant motion vectors were reused for the chrominance samples. After that, these ME algorithms were performed for the three channels (Y, Cb and Cr). Used videos uses a chrominance subsample of 4:2:0, so using 16x16 size blocks for the ME at luminance samples, the block size used by the ME on the chrominance samples is 8x8 .

The results presented in Table 1 were generated from software simulation of ten HD 1080p test sequence [10]: blue_sky, man_in_car, pedestrian_area, rush_hour, station2, sun_flower, riverbed, rolling_tomatoes, traffic and tractor. Each sequence have different characteristics, thus it is able to evaluate all distinct particularities, such as high and low motion activity.

As it is able to be observed in Table 1, the quality results of the proposed approach are shown in Δ PSNR (dB), i.e. ME gains in each chrominance channel when compared with reuse of the motion vectors for the chrominance channels. Likewise, the computational cost is evaluated in Δ CBC $x10^8$, which represents the computational cost increase generated by this new approach.

As it is able to notice from Table 1, despite the increasing in terms of computational cost, this work shows significant gains in each chrominance channel for all evaluated sequences.

Comparing all results it is possible to notice that the average quality gain for Cb and Cr channels were 2.385dB and 3.022dB, respectively. Analyzing the computational cost, it is possible to realize that it was increased 1.5 times. Since PSNR is evaluated in a logarithmic scale, one dB of variation at PSNR generates a significant gain, or loss, in the video image quality.

A. Evaluation focusing on hardware implementation

Pixel sub sampling is large used by ME in order to accelerate the SAD calculation process [11]. This technique can be used without generating expressive quality loss. When sub sampling is set to 2:1, only the half of block pixels will be compared. Likewise, with sub sampling 4:1, a quarter of block pixels is considered in the comparisons performed by SAD calculations.

This technique is mainly valid for hardware implementations, since the amount of data to be processed is reduced expressively. Thus, sub sampling reduces the number of comparisons done, what simplify SAD calculation.

Another important technique to be evaluated is the restriction at iteration numbers. It happens due the fact that the fast algorithm development in hardware for ME has a lot of troubles. The impossibility to determinate how much iteration will be necessary to generate a motion vector is one of the main problems. Thus, it is impossible to evaluate how many clock cycles are necessary to the architecture deliver a motion vector [5].

Some tests were done in order to check the impact in terms of objective quality and computational cost when sub sampling and restriction on iteration number are used. Table 2 shows average results for in software simulation, using ten HD 1080p test sequences and the number of iterations were restricted to five. The sub sampling levels evaluated was 2:1 and 4:1. The results presented in Table 2 represent average results for the first 100 frames of each previously presented video sequence.

The quality results are presented in PSNR, and the computational cost in number of compared pixel (NCP). The use of pixel sub sampling makes unfair the comparison only about the number of compared blocks. A block of 16x16 with 2:1 sub sampled has only half of the total number of pixels, and only a quarter when using 4:1 sub sampling. So, the computational cost to process one candidate block is significantly different.

As it is able to be observed in Table 2, the MPDS algorithm, applied also on chrominance samples – with 2:1 sub sampling and iteration restriction – achieves a quality average gain of 2.041dB (Cb channel) and 2.524dB (Cr channel). These gains are obtained when comparing to the original algorithm, which reuses motion vectors. It is also notable an average decreasing of NPC in $66{,}05 \times 10^8$.

On the other hand, the version of the same algorithm with 4:1 sub sampling has obtained a lower average gain, achieving 1.257dB in Cb and 1,767dB in Cr, using the same comparison. Even being lower, these gains are also expressively. The computational complexity had an average decreasing of $172{,}90 \times 10^8$ NPC.

Table II.
Quality and computational cost results for the MPDS algorithm on chrominance samples

		MPDS Y/Cb/Cr	MPDS [5]	MPDS Y/Cb/Cr / 2:1 5 iterations	MPDS Y/Cb/Cr / 4:1 5 iterations
PSNR	Y	34,791	33,280	33,612	33,612
	Cb	40,367	-	42,409	41,625
	Cr	42,285	-	44,810	44,052
Δ PSNR	Y	-	-	-	-
	Cb	-	-	2,041	1,257
	Cr	-	-	2,524	1,767
NCP $\times 10^8$		272,64	80,128	206,592	99,744

Therefore, sub sampling is an interesting alternative for high definition videos when performance restriction is a priority. Although the quality losses are unexpressive, the decreases in terms of cost computation are significantly mainly compared to original version presented in Table 1. Thus, it results in a higher hardware performance.

IV. ARCHITECTURAL PROPOSAL FOR CHROMINANCE MOTION ESTIMATION

The proposal architecture for this work is based on a previous developed architecture of MPDS for luminance samples, presented in [5]. This algorithm shows an important feature for its hardware development which is the possibility of parallel execution. Basically, it is used five search cores where each one process the DS algorithm in parallel, making the performance of this architecture almost equal to a DS core [5].

As it was evaluated before (Chapter III, section A), MPDS implementation was done using restriction in the number of iterations and sub sampling as well. Thus, the architecture can achieve a higher performance, facilitating the integration with other encoder modules [5]. In the MPDS architecture for luminance samples, it was done the 4:1 sub sampling and 16x16 blocks size. In this work, it is used 4:1 sub sampling for 8x8 block size of chrominance samples. The number of iterations was fixed to five in the both of architectures.

In the proposed architecture, there are two modules. The first one process the luminance information while the other one process the chrominance information. For each architecture module of MPDS algorithm, it is used five cores of DS architecture. One core is responsible to process the central region and the other ones for each sector of the search area. Figure 1 illustrates a block diagram of the developed architecture for MPDS algorithm.

The architecture hierarchy is relatively simple, because each module works independently. Firstly, the module Y (luminance) starts its processing together with other module (Cb/Cr), which begins to process Cb samples. After Cb samples are processed, the module two begins to process Cr samples. It occurs due the fact that the Y blocks size is 16x16 and the Cb/Cr blocks size is 8x8. Thereby, the processing of Cb and Cr channels can be done sequentially, reusing the same hardware.

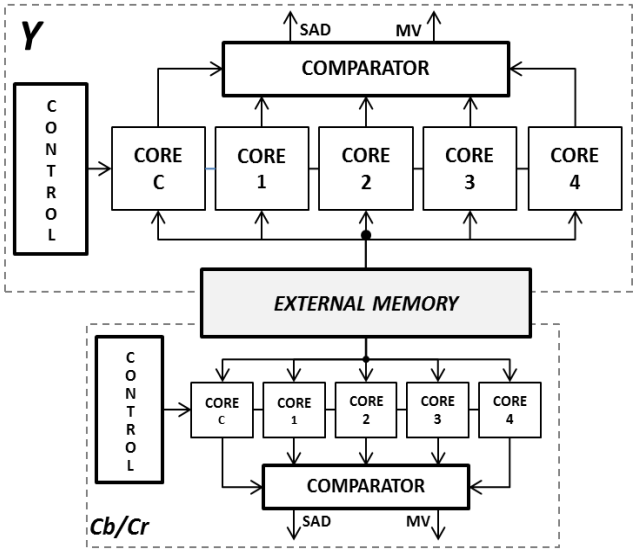

Fig. 1 Proposed Architecture

Analyzing each module, it is possible to realize that they perform the same process. Inside the modules, each core also works in an independently way. Thus, after their searches enclosing, each core write its result at the comparator that defines which core found the lowest result for the SAD. The processing on the module starts filling the reference memory of the Core C. After the reference memory is filed, the C core is started. This process is repeated for the other cores until the reference memory of core 4 is filled. Thus, after the processing of core 4, the comparator verifies which is the best candidate, and generates a movement vector.

Based on MPDS architecture for luminance described at [5], it was done an estimative for memory and processing cycles. As it is possible to be observed in Table 3, each architecture module has 5 processing cores, totaling 10 processing cores. Each luminance core has 16.4 Kbtis of internal memory, thus altogether this module has 82Kbtis of memory. On the other hand, in the chrominance module, each core has 4.1Kbits of memory, totaling 20.5Kbits. Thereby, the architecture has 102.5Kbtis of internal memory. Moreover, analyzing Table 3 is also possible to realize the luminance module defines the total number of cycles per block (170 cycles per block). Since the original MPDS architecture is able to process 145 HD 1080p frames, this purposed architecture is also able to process the same resolution in real time.

Table III.

Architecture proposal for the MPDS algorithm for luminance and chrominance information

Architecture	Number of Cores	Memory per Core	Memory size	Cycles per Block
MPDS Luminance	5	16,4 Kbits	82 Kbits	170
MPDS Chrominance	5	4,1 Kbits	20,5 Kbits	80

V. Conclusion and Future Works

This paper presented a new approach for Motion Estimation focused on high definition video compression. This technique uses the chrominance information in order to achieve objective quality increasing. In this work, a hardware architecture for chrominance samples motion estimation, focused on high definition video real time processing was proposed. This architecture is based on a previous developed architecture for motion estimation on luminance samples which implements the MPDS algorithms. From this technique, it was possible to reach an average gain in image quality of 2.385dB in Cb and 3.022dB in Cr chanels. On the other hand, this aproach increases the computational cost 1.5 times.

Since the architecture which this work was based consumes 82Kbits of memory whilst the proposed architecture needs 102.5Kbits, there is just 25% of memory increasing. Moreover, analyzing the purposed architecture, it is possible to infer that it will able to process HD 1080p videos in real time (30 frames per second).

As future works, it is intended to evaluate the emerging standard HEVC in order to apply the Motion Estimation using chrominance samples. Moreover, it is intended to describe the proposed architecture and synthesize it for FPGAs.

VI. References

1] Cheng, Y, "An H.264 Spatio-Temporal Hierarchical Fast Motion Estimation Algorithm for High-Definition Video," *IEEE ISCAS*, pp. 880-883, 2009.

[2] G. Sullivan, A. Luthra T. Wiegand, "Draft ITU-TRecommendation and final draft international standard of joint video specification," *ITUT Rec.H.264|ISO/IEC 14496-10 AVC*, 2003.

[3] R. GONZALEZ and R WOODS, Processamento de Imagens Digitais. São Paulo: Edgard Blücher, 2003.

[4] Agostini, L. V., Desenvolvimento de Arquiteturas de Alto Desempenho Dedicadas à Compressão de Vídeo Segundo o Padrão H.264/AVC. Universidade Federal do Rio Grande do Sul [manuscrito], 2007.

[5] Porto, Marcelo, Desenvolvimento Algorítmico e Arquitetural para a Estimação de Movimento na Compressão de Vídeo de Alta Definição [manuscrito], 2012.

[6] I. RICHARDSON, *H.264/AVC and MPEG-4 Video Compression – Video Coding for Next-Generation Multimedia*, John Wiley and Sons ed. Chichester, 2003.

[7] S. et al ZHU, "A new Diamond Search Algorithm for Fast Block-Matching Motion Estimation," *IEEE Transactions on Image Processing*, 2000.

[8] SANCHEZ, G., et al, "A Real Time HDTV Motion Estimation Architecture for the New MPDS Algorithm," *IEEE EUROCON*, 2011.

[9] SANCHEZ, G., et al, "Spread and Iterative Search: A High quality Motion Estimation algorithm for high definition videos and ITS VLSI design.," *IEEE ICME*, 2012.

[10] XIPH.ORG. (2010) Xiph.org Test Media. [Online]. <media.xiph.org/video/derf/>

[11] KUHN, P, Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation, 1999, Boston: Kluwer Academic Publishers.