Multi-Level Pipelined Parallel Hardware Architecture for High Throughput Motion and Disparity Estimation in Multiview Video Coding

Bruno Zatt†‡, Muhammad Shafique†, Sergio Bampi‡, Jörg Henkel†
†Karlsruhe Institute of Technology (KIT), Chair for Embedded Systems, Karlsruhe, Germany
‡Federal University of Rio Grande do Sul (UFRGS), Informatics Institute/PGMICRO, Porto Alegre, Brazil
{bzatt, bampi}@inf.ufrgs.br, {muhammad.shafique, henkel}@kit.edu

Abstract

This paper presents a novel motion and disparity estimation (ME, DE) scheme in Multiview Video Coding (MVC) that addresses the high throughput challenge jointly at the algorithm and hardware levels. Our scheme is composed of a fast ME/DE algorithm and a multi-level pipelined parallel hardware architecture. The proposed fast ME/DE algorithm exploits the correlation available in the 3D-neighborhood (spatial, temporal, and view). It eliminates the search step for different frames by prioritizing and evaluating the neighborhood predictors. It thereby reduces the coding computations by up to 83% with 0.1dB quality loss. The proposed hardware architecture further improves the throughput by using parallel ME/DE modules with a shared array of SAD (Sum of Absolute Differences) accelerators and by exploiting the four levels of parallelism inherent to the MVC prediction structure (view, frame, reference frame, and macroblock levels). A multi-level pipeline schedule is introduced to reduce the pipeline stalls. The proposed architecture is implemented for a Xilinx Virtex-6 FPGA and as an ASIC with an IBM 65nm low power technology. It is compared to state-of-the-art at both algorithm and hardware levels. Our scheme achieves a real-time (30fps) ME/DE in 4-view High Definition (HD1080p) encoding with a low power consumption of 81 mW.

1. Introduction and Related Work

The increasing trend of devices with 3D-video displays stimulates the need for 3D video (based on multiple view[1] sequences) encode and decode features. The 3D personal video recording is envisaged to be the key application for next-generation mobile devices [1]. Such devices with 2-views 3D recording [2][3] have recently been released. However, for the next-generation 3D devices more than 2 views (4-8 views) are expected, which poses serious challenges on the real-time 3D video encoding.

The Multiview Video Coding (MVC) standard [5] has been introduced recently to provide 20-50% additional bit-rate reduction [1] compared to H.264/AVC. This improved compression comes at the cost of increased complexity due to multiple block-sized Motion and Disparity Estimation (ME, DE). The DE eliminates the redundancy between different views, while the ME eliminates the redundancy between different temporal frames. The high complexity of ME and DE makes MVC encoding infeasible for the real-time applications, especially when considering the battery-powered devices. Fig 1 presents the coding time distribution of the MVC encoder for one Group of Pictures (GOP, see Section 2.2) of *Ballroom* sequence (640x480, with high motion). On average, the ME/DE contributes up to 98% of the total coding time for one whole GOP (up to 99% for some views). Note, a fast ME/DE estimation (*TZ Search* [4]) is considered in these experiments, which is up to 23x faster than the *Full Search* (exhaustively search all possible candidate positions) [4].

The key challenge is, how to expedite the whole ME and DE process in order to meet the real-time MVC encoding requirements for multiview High Definition HD1080p (1920x1080, a typical resolution for personal video recording in 3D-camcoders, for instance). State-of-the-art approaches propose either fast ME/DE algorithms or hardware acceleration to obtain high throughput.

State-of-the-art fast ME/DE algorithms employ variable search range based on disparity maps [7] or considering the cameras geometry [8]. A fast direction prediction (ME or DE) based on the blocks motion intensity is proposed in [9]. In [10] a fast ME based on *complete* DE is proposed. The view-temporal correlation is exploited in [11] by using the motion and disparity information of the first encoded view in order to reduce the computational complexity of further views. The inter-view correlation is also evaluated in [12] to reduce ME/DE complexity. Algorithm and architecture for disparity estimation with mini-census adaptive support is proposed in [13]. The main drawback of these fast ME/DE algorithms resides in the fact they do not exploit the full potential of the 3D-neighborhood correlation available in spatial, temporal and disparity domains. Moreover, even the more sophisticated techniques are dependent on the complete first view encoding. By encoding one view, the motion field information can be extracted but not the disparity field information (as no inter-view prediction is performed in this case). Therefore, it potentially limits the speedup of disparity estimation. Although state-of-the-art ME/DE algorithms provide significant computation reduction[2], hardware accelerators are inevitable to realize real-time ME/DE in MVC encoding on the mobile devices.

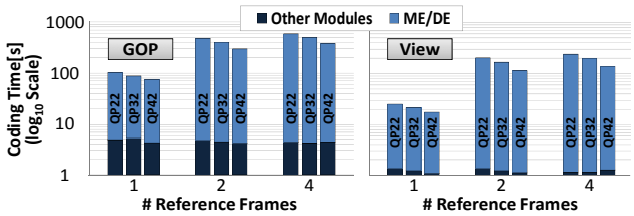

Fig 1. Coding time (a) for one complete GOP; (b) for one GOP of view 1 (Ballroom, *TZ Search* [4], [±96, ±96])

The most prominent related work on the hardware design of ME/DE is proposed in [14][3]. This work implements a cache-based integer ME/DE claiming to support real-time MVC encoding for four 720p views. This architecture uses a predictor-centered [±16,±16] *Full Search*. However, the *Full Search* requires a high number of SAD operations per Macroblock (MB, 1089 candidates per prediction direction) which limits the throughput and incurs significantly higher power (see Section 4.2). Therefore, there is a dire need to *address the issue of high ME/DE complexity in MVC at both algorithm and hardware levels* in order to obtain real-time multiview HD1080p ME/DE in MVC.

1.1. Our Novel Contributions

We propose a high-throughput ME/DE scheme for MVC that integrates a fast ME/DE algorithm with a multi-level pipelined parallel hardware to expedite the ME/DE process jointly at the algorithm and hardware levels. In particular, our novel contributions are:

1) A fast ME/DE algorithm that computes the confidence of predictors (motion/disparity vectors of the neighboring MBs) in the 3D-neighborhood to skip the search step
   - the search pattern is replaced by the SAD calculation for a reduced number (up to 15) of predictors (obtained by performing an in-depth analysis of the 3D-neighborhood)
2) A multi-level pipelined parallel hardware architecture for ME/DE that exploits the **four levels of parallelism** available in the MVC encoder, i.e., view, frame, reference frame, MB levels (see details in Section 3.2). The proposed ME/DE architecture employs:
   - two parallel control and data prefetch modules

[1] A group of video sequences captured by different cameras in a 3D-scene.

[2] Besides fast mode decision schemes, like the one proposed in [21].

[3] Also used for comparison in Section 4.

978-3-9810801-7-9/DATE11/©2011 EDAA

- a shared array of 64 4-pixel SAD accelerator and adder trees
- a novel multi-level pipeline schedule
- a memory hierarchy with 3 MB-level data caches, caches for motion and disparity vectors, and 2 Address Generating Units.

The proposed hardware architecture facilitates the integration of different data prefetching schemes (e.g. based on [15] and [17]) to reduce the external memory bandwidth by up to 65%.

To the best of our knowledge, this is the first multi-level pipelined parallel architecture for joint ME/DE in MVC that – besides four levels of parallelism at the hardware level – exploits the 3D-neighbor-hood correlation at the algorithmic level to provide real-time ME/DE in MVC for four HD1080p views.

This paper is organized as follows: Section 2 presents the proposed fast ME/DE algorithm and the 3D-neighborhood analysis. The multi-level pipelined hardware architecture and its scheduling are presented in Section 3. Section 4 discusses the results and comparison with state-of-the-art followed by the conclusion in Section 5.

2. Fast Motion/Disparity Estimation Algorithm

2.1. Basic Prediction Structure of MVC

Before proceeding to our novel algorithm, we briefly describe the basic prediction structure of MVC to a level of detail necessary to understand the novel contribution of this paper. The MVC uses the motion and disparity estimation tools to eliminate the temporal and view redundancies between frames, respectively. The prediction structure used in this paper is presented in Fig 2. The *'I frames'* are intra-predicted frames (i.e., no ME/DE is used), *'P frames'* use unidirectional prediction or estimation (in this example the *'P frames'* use only DE in one direction), and *'B frames'* use bidirectional prediction having reference frames in at least two directions. The arrows represent the prediction directions; frames at the tail side act as reference frames to the frames pointed by the arrowheads. Note that some frames have up to four prediction directions. In order to provide access points, the video sequence is segmented in Groups of Pictures (GOPs) where the frames located at the GOP borders are known as *Anchor* frames and are encoded with no reference to the previous GOP. All other are called *Non-Anchor* frames.

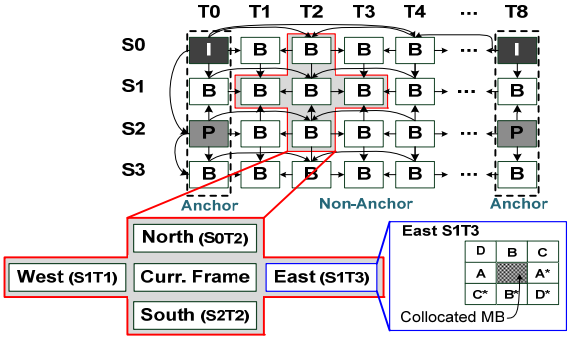

Fig 2. MVC Prediction structure and 3D-Neighborhood details

2.2. Analyzing the Motion and Disparity Vectors in 3D-Neighborhood

The same objects in a 3D scene are typically present in different views (except for occlusions). Consequently, the motion perceived in one view is directly related to the motion perceived in the neighboring views [11]. Moreover, considering parallel cameras, the motion field is similar in these views [8]. Analogously, the disparity of one given object perceived in two cameras remains the same for different time instances when just translational motion occurs. Even for other kinds of motion the disparity is highly correlated. Similar observations have been carried out by [8][11][12].

Based on these observations, an offline analysis of the motion and disparity vectors (MV, DV) in the 3D-neighborhood is performed to quantify the difference between the predictors and the optimal vector[4], i.e., MV/DV error. A set composed of 1 *spatial* median predictor, 6 *temporal* predictors and 6 *disparity* predictors is analyzed. The temporal predictors are selected from the previous and next frames (in the displaying order) called *West* and *East* neighbor frames, respectively. For each neighboring frame, three predictors are calculated (see Fig 2): (a) the collocated MB (MB in the reference frame with the same relative position of the current MB), (b) median up (using A, B, C, D as shown in Fig 2: see details in standard [5]), and median down (using A*, B*, C* and D*, see Fig 2). The disparity predictors from the *North* and *South* neighboring view frames are obtained by considering the Global Disparity Vector (GDV).

Fig 3 illustrates the MV/DV error distribution for *Vassar* (low motion) and *Ballroom* (high motion) sequences (a subset of the sequences recommended by JVT for multiview video testing [6]) in the 3D-neighborhood. Each plot represents the difference between a given predictor (in this case for the spatial predictor and 3 collocated predictors in different neighboring frames) and the optimal vector of the current MB. It shows that for majority of the cases, the predictor vectors have similar values in comparison to the optimal vector. Even, most of the predictors have exactly the same value of the optimal vector. Our analysis shows that this observation is valid for the other nine predictors in all direction of the 3D-neighborhood as depicted in Fig 3 (only few error plots are shown here due to the page limit).

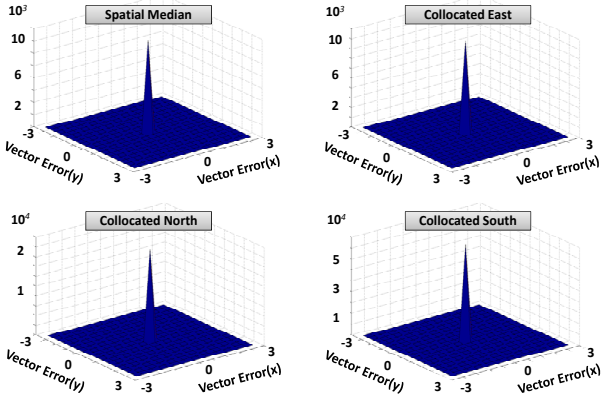

Fig 3. MV/DV error distribution in Ballroom and Vassar for different predictors compared to the optimal MV

TABLE I. VECTOR HITS FOR DIFFERENT PREDICTORS (BALLROOM)

Predictor	Neighbor Frame	Hit [%]	Available [%]		Neighbor Frame	Hit [%]	Available [%]
Spatial	**n.a.**	94.12	99.90	-	-	-	-
All	***West***	96.94	51.52	**Median Up**	***West***	54.74	99.90
	East	97.93	60.30		***East***	63.78	99.90
	North	97.94	65.40		***North***	93.17	73.99
	South	98.67	21.29		***South***	94.61	23.98
Collocated	***West***	58.43	99.90	**Median Down**	***West***	54.99	99.89
	East	66.79	99.90		***East***	63.92	99.89
	North	95.39	72.39		***North***	93.21	74.13
	South	96.75	23.48		***South***	94.70	23.93

To quantify the MV/DV error distribution in the 3D-neighborhood, several experiments were carried to measure the frequency of a given predictor that is equal to the optimal vector (i.e., $MV_{Pred} = MV_{Curr}$). When this condition is satisfied, it is denoted as a so-called *hit*. A set of different conditions was defined including, for example, the case when all predictors of a given neighbor frame (collocated, median up and median down) are *hits*. Table I presents the detailed information on the vector *hits* where the *Availability* is the percentage of cases when that predictor is available. The disparity predictors present the higher number of *hits* followed by the spatial and temporal predictors. Considering the quality of predictors in the same neighboring frame, the collocated predictors present better results in relation to median up and median down. The latter two present similar number of *hits*. In the case when *all predictors of a given neighboring frame are availa-*

[4] Obtained with Full/Exhaustive Search algorithm.

ble, the predictor is highly accurate providing up to 98% *hits* (at least one among all predictors is equal to the optimal MV/DV).

In conclusion, there is a high vector correlation available in the 3D-neighborhood that can be exploited during the ME/DE. Once the predictors point to the same region as the optimal vector, there is no need for search patterns over a big search range. Moreover, for most of the cases the predictors' accuracy is high enough to completely avoid the ME/DE search and refinement stages (see Section 4).

2.3. Fast ME/DE Algorithm

Our fast ME/DE algorithm is based on the above-presented analysis of the MV/DV error distribution in the 3D-neighborhood. To exploit this knowledge, accurate motion and disparity fields must be available. Therefore, at least one frame using DE and one using ME must be encoded with the optimal or a near-optimal searching algorithm. In our scheme, to avoid a significant quality loss, all *Anchor* frames and the frames situated in the middle of the GOP are encoded using the *TZ Search* algorithm [4] (used as base for our solution since it is the key fast ME/DE algorithm used in JMVC [16]). The *Anchor* frames are encoded using DE, while the frames in the middle of a GOP use ME and/or DE according to the view they belong. The frames encoded with high effort are referred as *Key Frames* (KF), while the others are the *Non-Key Frames* (NKF). The KF are defined as being the first and second temporal prediction layers[5] of the MVC prediction structure [1] to avoid propagating the prediction error in the upper layers. Once the motion and disparity fields are established all NKF can be encoded based on predictors available in these fields. Our fast ME/DE algorithm skips the entire ME/DE search pattern stage for all NKF by using the predictors from the 3D-neighborhood. It provides a significant computation reduction compared to the *TZ Search* algorithm (see Section 4).

Fig 4 presents the operational flow of our fast ME/DE algorithm for the NKF coding. For KF the *TZ Search* is used (as discussed earlier) to provide good a motion/disparity field and it is not shown in the figure.

The algorithm operates in three main phases:

Phase-1: Frame Level MV/DVs evaluation: First, the *Availability* of the temporal and disparity predictors is checked. Using the available predictors, the current MB is pre-classified into one out of the two prediction groups: *Ultra Fast Prediction* and *Fast Prediction* based on the predictors' *Confidence Level*. Note that the spatial predictor is not used in this phase, as it is not available at the time of this decision due to the spatial dependencies. The *Confidence Level* of each predictor is calculated as the sum of the *hits* (i.e., $CL_{Pred} = \Sigma Hits_{NeighborX}$) of all predictors with the same vector. The CL is based on the offline-analysis presented in Table I. If one predictor has a *Confidence level* bigger than a threshold (i.e., $CL_{Pred} > CL_{TH}$), it is set as *Common Vector* and the current MB is classified to be encoded using the *Ultra Fast Prediction*. Otherwise, the MB is classified to be encoded using the *Fast Prediction*. The threshold is defined using offline statistical analysis and formulated as a function of the Quantization Parameter (QP) using polynomial curve fitting.

Phase-2: MB Level MV/DVs Evaluation and Prediction: In this phase, the spatial predictor may be formulated considering the raster scan order of MB encoding. Therefore, the MB classification done in the previous phase is further refined using the knowledge of the spatial predictor. In case of *Ultra Fast Prediction*, only three vectors are evaluated for each reference frame. These are the predictor with the highest *Confidence Level* (also referred as *Common Vector*), the Zero vector, and the SKIP vector (median predictor in P frames and direct predictor in B frames [5]). The Zero and SKIP vectors are evaluated due to their high occurrence probability. In case of *Fast Prediction*, all of the available predictors are evaluated in addition to the Zero and SKIP vectors. In the worst case, only 15 candidates are evaluated in each reference frame for a MB, which is still a much smaller number compared to the number of candidates processed by state-of-the-art (on average 44 SADs by UMHexagonS and 23 by EPZS [18][22]).

[5] The hierarchical prediction structure defines the coding order and how the frames refer to the others. A good coding quality for the first layers is of key importance as the other layers use them as reference.

Phase-3: MV/DV Storage: At the end, the best MV and/or DV are stored, which will be used for the prediction on the later MBs according to the prediction structure as shown in Fig 2.

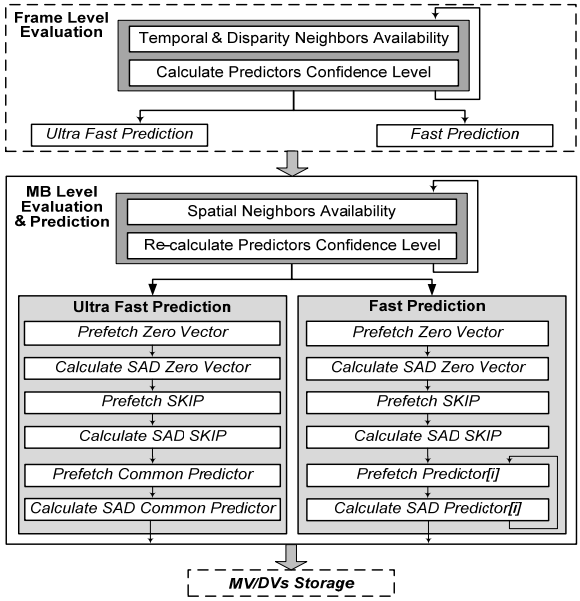

Fig 4. Operational Flow of our fast ME/DE algorithm

3. Multi-Level Pipelined Parallel Hardware Architecture for Motion and Disparity Estimation

As previously discussed, although our fast ME/DE algorithm provides significant computation reduction (see Section 4), a high throughput hardware is required for real-time ME/DE encoding on mobile devices. Therefore, we have integrated the fast ME/DE algorithm in our multi-level pipelined parallel hardware architecture to provide real-time ME/DE for up to 4 views HD1080p. It employs a novel pipeline scheduling technique that exploits the four levels of parallelism inherent to the MVC prediction structure as discussed in Section 3.2.

3.1. Multi-Level Pipelined ME/DE Hardware Architecture

Fig 5 shows the block diagram of our ME/DE hardware architecture. It is composed of three main modules: (a) TZ Search Module, (b) Fast ME/DE Module, and (c) Shared SAD calculator consisting of 64 SAD operators and adder trees. The *SKIP Vectors Prediction* module and the MV/DVs storage memory are also shared by the TZ and fast ME/DE modules. The blocks outside the filled region represent other MVC encoder modules, which are out of the focus of this work. The three main modules are explained in the following. For the simplicity of Fig 5, the CG logic is not shown here.

a) The TZ Search Module is composed of three sub-modules: (i) TZ Search Control, (ii) Address Generating Unit (AGU), and (iii) Cache Memory. The TZ Search Control requires the areas to be stored in the Cache Memory and the candidate blocks to be sent to perform the SAD calculation. The implemented cache memory exploits the regularity of *TZ Search* algorithm by using the Level-C data reuse scheme as described in [15]. The Level-C reduces the memory access by storing the search window for one MB and prefetching only the missing data for the next MB. By using this data reuse and prefetching technique, it is possible to reduce bandwidth and meet real-world memory constrains.

b) The Fast ME/DE Module differs from the TZ module in its control logic and the cache memory organization. The control logic is responsible for (i) reading the MV/DVs from a dedicated memory, (ii) calculating the Common Vector (in case the *Ultra Fast Prediction* is used), and (iii) to fetch the reference frame data. Since the fast ME/DE module evaluates candidate blocks with very different regularity behavior, the cache is composed of two small memories. Since the Zero Vectors show regular distribution,

the Level-A caching technique [15] is used for the Zero Vector Cache. The Level-A performs predictor prefetching that guarantees the data availability for a pre-determined predictor such as Zero Vector. For the 3D-neighborhood predictors, access to different reference frame area may be required. For this reason, our architecture employs the 3D-Cache [17] originally developed for the H.264 decoder motion compensation. In our ME/DE the 3D-cache is composed of 16 sets instead of the original 32 considering that our solution evaluates the predictor just for one reference frame at a given time instance. SKIP data also uses this cache memory. Since the 3D-cache technique does not prefetch the complete search window, it requires less memory bits compared to a Level-C based prefetching. Due to the data locality, the 3D-cache technique provides an average cache hit of 80%. Considering that the fast ME/DE module do not requires high throughput and works in parallel to TZ module, 20% cache misses do not represent a performance losses.

c) **Shared SAD Calculator:** The relation between the processing time for one complete *TZ Search* and one Fast ME/DE algorithm flow is about 1:100. Therefore, to balance the parallel utilization of the TZ and fast ME/DE modules, the TZ module throughput needs to be kept 100x higher than that of the fast ME/DE module, thus demanding a significantly bigger hardware. Alternatively, a Shared SAD Calculator hardware is proposed as a compromise that guarantees the full usage of the SAD operators (and the adder trees) throughout the ME/DE process. If the fast ME/DE is performed some operators are allocated for this module, otherwise all SAD operators are allocated to the TZ module. This solution allows exploring the parallelism by simply varying the number of SAD operators (and adder trees) and the SAD operator allocation. In order to obtain 4 views HD1080p real-time, 64 4-pixel SAD operators and 11 adder trees are instantiated.

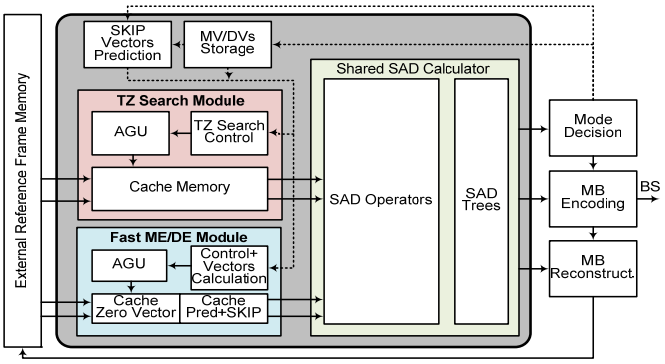

Fig 5. ME/DE Hardware Architecture Block Diagram

3.2. Multi-Level Pipelined Parallel Scheduling

Due to the prediction structure used in the MVC (see arrows in Fig 6b), four levels of parallelism can be exploited to achieve high throughput. For the ease of understanding, the frames in Fig 6b are ordered according to the coding sequence using numbers for the KF and the alphabet order for NKF. The *'I'* frames are not processed by ME/DE and are considered available. Frames 2, 4 and 6 are considered available from the previous GOP. The four levels of parallelism are:

i) *View-Level Parallelism:* Although MVC standard recommends a "Time First" decoding order[6], it is not mandatory to complete the encoding of the first view to start encoding the second view. For instance, S2 can be encoded in parallel to S0. In parallel to frame 1, frame 4 can be encode. Moreover, after encoding frame 1, frame 5 can be processed. Therefore, the ME and DE of the frames of different views can be processed in parallel.

ii) *Frame-Level Parallelism:* Within a view, there are frames with no dependencies between them. For example, using one reference frame per prediction direction, frames A and B can be processed in parallel. Similarly, it is possible to process C, D, E, and F in parallel.

iii) *Reference Frame-Level Parallelism:* Every single frame can be predicted using four prediction direction. The search in different prediction direction has no data dependencies and allows for parallel processing. For instance, Frame P uses references from frames 5, D, M and J, that can be processed in parallel.

iv) *MB-Level Parallelism:* Each MB has data dependencies to the previous one because of the spatial median predictor and SKIP vector. However, using the fast ME/DE algorithm (see Section 2.3) it is possible to start the predictors evaluation without the spatial one (Frame Level Evaluation), prefetch the data and calculate SAD before having the previous MB results (it is also possible for the Zero Vector).

A novel scheduling scheme is proposed that exploit the above-discussed four levels of parallelism (see Fig 6a). The numbers and letters are consistent to Fig 6b. The letters between the parentheses represent the prediction direction E, W, N, and S. The dotted boxes represent a frame that belongs to the next GOP. This scheduling assumes the existence of two hardware modules (see Fig 5) operating in parallel: (i) one processing the *TZ Search* for KF, and (ii) one processing the fast ME/DE for NKF.

Each time slot of the TZ Module is dedicated to perform the search for a complete KF in one prediction direction. It is noted that the average coding time for the prediction structure of Fig 6b (4 views each with a GOP size=8) is the time to perform 16 TZ searches. This corresponds to a reduction of 81% in the number of total TZ searches when compared to a system **without** our fast ME/DE algorithm (that performs 88 complete TZ searches).

After the required KF are processed by the TZ module (considering the data dependencies), all NKF in the same view can be processed in a burst following the coding order (as shown by the alphabetic order in Fig 6a). The data dependencies between the KF and NKF are represented in Fig 6a by dashed arrows. To avoid stalls, the GOP-level pipeline starts the *TZ Search* in KF of next GOP while the fast ME/DE module concludes the current GOP processing. Since the fast ME/DE represents less than 1% of the processing effort of *TZ Search* the fast ME/DE module remains idle for a relatively long time. Therefore, it is clock-gated (CG) till the next usage. For simplification, the details of prediction directions for the NKF are not presented in Fig 6a. However, for each NKF all of the required prediction directions are serially evaluated. For instance, in the slot of frame A, *West* and *East* directions are evaluated.

The internal scheduling (i.e., at the MB level) of the TZ Module presented in Fig 6c. The three main pipeline stages are:

i) TZ Search control: it determines the candidate point; this logic in hardware is always active

ii) TZ Search window prefetch: it prefetches the reference pixel data for the entire search window from reference memory; this logic in hardware is clock-gated after bringing the search window

iii) TZ SAD computation: it starts processing as soon the first useful reference block is available in the cache of the TZ Module

As the fast ME/DE algorithm has two prediction modes (see Section 2.3), two distinct pipeline schedules are proposed in Fig 6d and Fig 6e for the *Ultra Fast Prediction* and *Fast Prediction* modes, respectively. The three main pipeline stages are: (i) fast ME/DE Vector Calculation, (ii) data Prefetch, and (iii) SAD calculation. First, the Zero Vector is evaluated, as it has no data dependencies with the spatial neighbors. Afterwards, the predictors are evaluated and the Common Vector or Predictor Vector 1 are processed (represented by the grey blocks in Fig 6d and Fig 6e). If the spatial vector is different from the other predictors, additional data is prefetched for further processing (i.e., for Predictor Vector 2-N). This corresponds to the MB-Level Evaluation step of Fig 4. The last vector to be evaluated is the SKIP predictor that depends upon the previous MB. In this pipeline stage, the MV/DV information of the previous MB must already be available to avoid pipeline stalls. The borders of the MB-level pipeline

[6] In a GOP all frames of a given view are processed and then the next view.

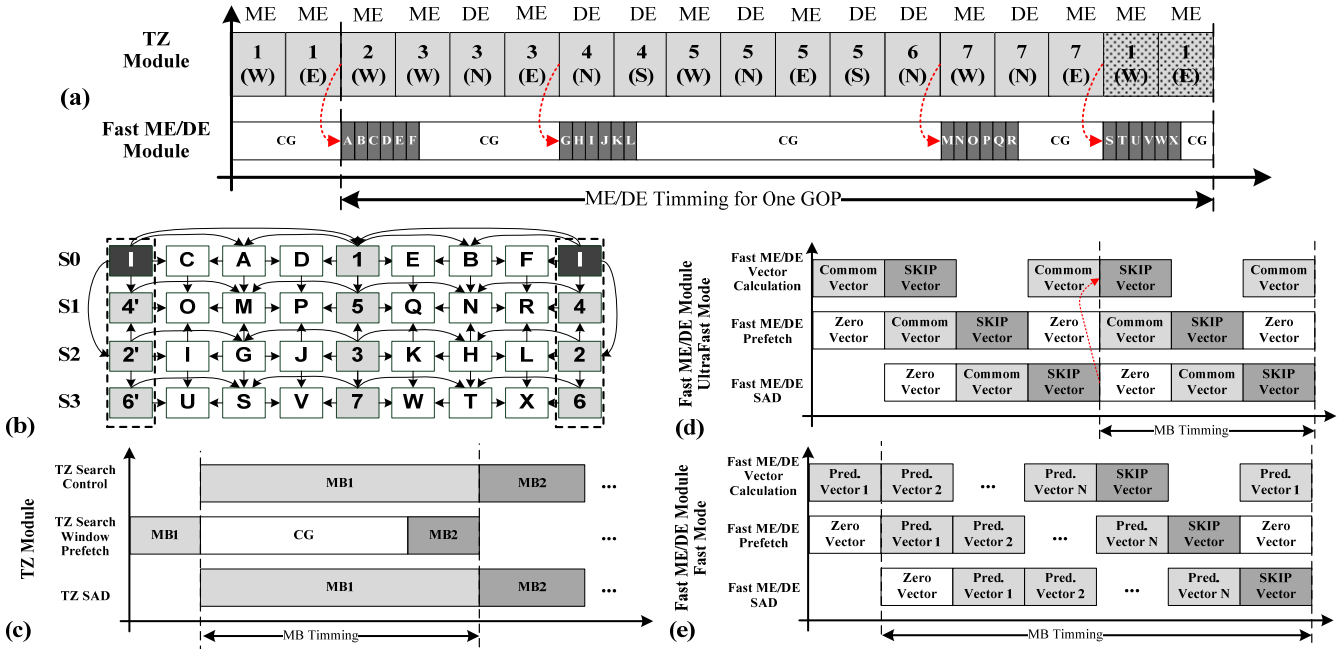

Fig 6. (a) GOP-Level Pipeline Schedule considering the (b) MVC prediction structure in our Fast ME/DE Scheme. MB-level pipeline schedule for (c) TZ Module and Fast ME/DE Module in (d) Ultra Fast and (e) Fast operation modes.

stage (indicated by the vertical dashed lines in Fig 6d and Fig 6e) are the same for both prediction modes allowing the mode exchange (Fast↔Ultra Fast) without any pipeline stalls.

4. Results and Evaluation

4.1. Evaluation of Fast ME/DE Algorithm

The fast ME/DE is evaluated using the JMVC 6.0.3 [16] framework. For the Key Frames, the *TZ Search* algorithm with a search range of [±64, ±64] is used for both ME and DE. The experiments are performed on a PC with an Intel Core 2 Duo-6600@2.4GHz with 2GB DDR2 memory and Windows XP SP2 operating system.

TABLE II. COMPARISON OF OUR FAST ME/DE ALGORITHM TO TZ SEARCH

Video	QP	*TZ Search* [16]			Fast ME/DE		
		Time [sec]	BR [kbps]	PSNR [dB]	TS [%]	ΔBR [%]	ΔPSNR [dB]
Ballroom (640x480) High motion	22	215.1	3298.026	40.709	85.9	8.4	0.011
	32	175.7	651.640	35.119	86.2	11.7	0.060
	42	127.1	188.178	29.318	84.9	19.8	0.190
Vassar (640x480) Low motion	22	171.1	3415.744	40.456	83.0	1.1	0.010
	32	110.0	315.142	35.226	82.4	6.7	0.013
	42	72.1	64.888	30.563	79.7	6.7	0.043
Breakdancers (1024x768) High motion	27	583.5	5680.204	41.172	86.0	15.7	0.087
	32	384.6	788.800	38.010	85.1	14.6	0.275
	42	262.9	277.970	33.803	82.8	8.9	0.304
Uli (1024x768) Low motion	22	600.9	12245.676	39.819	82.9	9.3	0.053
	32	516.8	2949.870	34.960	83.0	13.7	0.134
	42	406.5	849.462	28.944	82.4	8.5	0.191
Average	22	392.7	6159.913	40.539	84.5	9.6	0.040
	32	296.8	1176.363	35.829	84.1	13.1	0.121
	42	217.1	345.125	30.657	82.6	10.0	0.182
	Avg.	302.2	2560.467	35.675	83.9	10.9	0.114

In Table II the fast ME/DE results are detailed for the four evaluated sequences (this video sequences are recommended by JVT in the multiview test conditions recommendation [6]) considering three different QPs (22,32,42). The *TZ Search* is used for comparison as it is used for the Key frames and performs 23x faster compared to the *Full Search* (not used for performance comparison), while providing the similar rate-distortion (RD) results [4]. Compared to the *TZ Search*, our fast ME/DE provides 83% execution time saving at the cost of 11% increase in bitrate and 0.114dB of PSNR loss. In the best case, the execution time savings go up to 86%, which represents a significant computation reduction.

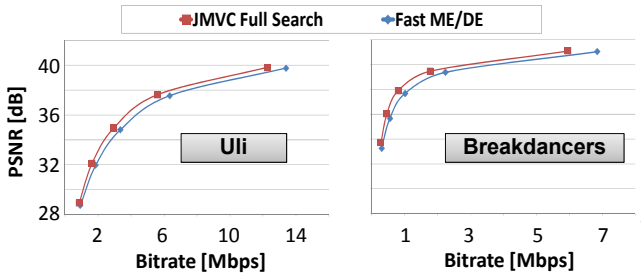

Fig 7. Rate-Distortion Comparison with Full Search

Fig 7 presents the RD-curves for two HD video sequences Uli and Breakdancers. It is noted that the RD curves of our fast ME/DE algorithm are close to that of the *Full Search* (used for quality comparison only). Compared to the *Full Search*, our fast ME/DE algorithm suffers from an insignificant loss of 0.116dB (on average).

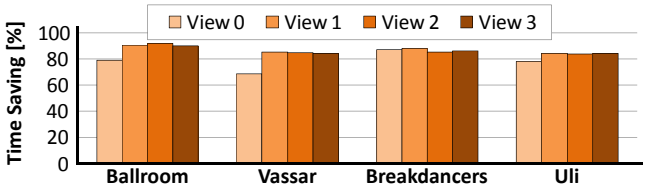

Fig 8. View-level execution time savings compared to *TZ Search*

The view-level execution time savings are presented in Fig 8. Note that for all views (except for View 0 of *Vassar* sequence) the time saving is ≥80%. The execution time savings for the high-motion sequences are slightly more than that in the low-motion sequences. Fig 9 presents the average number of SAD operations for ME/DE of one MB using *Full Search*, *TZ Search*, two state-of-the-art fast ME/DE algorithms presented in [14] and [19] (the technique proposed in [19] targets H.264 but is used here for comparison) and our fast ME/DE algorithm. Averagely, the proposed scheme reduces more than 99.9% in comparison to

Full Search, 86% to *TZ Search*, 99% and 94% compared to [14] and [19], respectively. Note, the results in Fig 8 and Fig 9 are for QP=32.

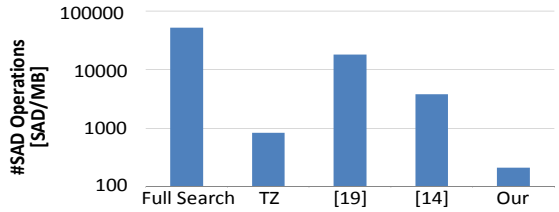

Fig 9. Comparison of the number of SAD operation

4.2. Evaluation of the ME/DE Hardware Architecture

The proposed architecture is designed, implemented, and tested in VHDL. It is synthesized and implemented (place and route) for a *Xilinx Virtex-6 xc6vlx240t* FPGA and an ASIC using *IBM 65nm LPe LowK* standard technology. The ASIC on-chip memory is based on the 1 Mb SRAM scheme for low power SoCs proposed in [20] that consumes 60mW at 300 MHz. The ASIC comparison to the ME/DE hardware of [14] and [13] are presented in Table III. The work in [13] requires more hardware resources with a much lower performance, targeting CIF (352x258) resolution. For the same operation frequency, our design is able to provide real-time ME/DE for up to four-views HD1080p videos compared to HD720p provided by [14]. This represents a throughput increase (in terms of the processed MBs) of 2.26x compared to [14]. Besides the faster hardware and the pipelined schedule considering four levels of parallelism, it is due to the reduced number of candidate blocks per MB provided by our fast ME/DE algorithm.

TABLE III. COMPARISON OF OUR FAST ME/DE ALGORITHM TO TZ SEARCH

	[13]	[14]	Fast ME/DE Architecture
Technology	UMC 90nm	TSMC 90nm Low Power LowK Cu	IBM 65nm LPe LowK
Gate Count	562k	230k	211k
SRAM	170 Kbits	64 Kbits	737 Kbits
Max. Frequency	95 MHz	300 MHz	300 MHz
Power	n/a	265mW, 1.2v	81mW, 0.8v
Proc. Capability	CIF @ 42fps	4-views 720p	4-views HD1080p

Memory Overhead: The increased memory is due to (a) the increased search range of [±64,±64] supported by our architecture, while [14] considers a [±16,±16] search area (i.e., 16x smaller), (b) the capability to encode HD1080p (thus support for more MBs), (c) support for three different caching schemes. Level-A requires 2 Kbits, Level-C 131 Kbits and 3D-cache 82 Kbits. Additionally, the MV/DV memory used by our fast ME/DE algorithm requires 522 Kbits (a limitation of our approach, but justified by high throughput). In relation to [14] our memory increases on 11.5x. Compared to [15] and [17], the three prefetching techniques jointly provide a external memory bandwidth reduction of up to 65%.

Area and Power Results: Our architecture requires 8% less gates compared to [14]. The power consumption (excluding the external memory accesses that are reduced by 65%) is also reduced by 69% (including the on-chip SRAM memory read/write power). However, it is important to consider that our architecture is implemented in 65nm at 0.8v while [14] uses a 90nm low power technology at 1.8v.

For the *Xilinx Virtex-6 xc6vlx240t* FPGA, our architecture processes real-time ME/DE for HD1080p at a maximum frequency of 258MHz. It requires 4,308 Slices (9,876 LUTS) and 103 BRAM modules.

5. Conclusion

We presented a multi-level pipelined parallel architecture for the motion and disparity estimation (ME/DE) in MVC encoding. The pipeline scheduling exploits the four levels of parallelism in the MVC prediction structure. The design employs a fast ME/DE algorithm that provides a computations reduction of 83% compared to the state-of-the-art *TZ Search* algorithm with insignificant quality loss (avg. 0.11dB). The hardware architecture throughput guarantees real-time MVC encoding for up to four HD1080p views.

The complete ME/DE system is implemented for an ASIC (using IBM 65nm low power technology) and an FPGA (Xilinx Virtex-6). The high throughput and low power results shows the practicability and industrial potential of our scheme for real-world solutions. The proposed work thereby enables real-time HD MVC encoding on mobile devices with low power.

6. Acknowledgment

We would like to thank Fabio Leandro Walter, Federal University of Rio Grande do Sul, for his contribution to the hardware implementation.

7. References

[1] P. Merkle et al., " Efficient Prediction Structures for Multiview Video Coding", Transaction on Circuits and Systems for Video Technology, vol.17, no.11, pp. 1461- 1473, 2007.

[2] Panasonic HDC-SDT750K: http://www2.panasonic.com/consumer-electronics/support/Cameras-Camcorders/Camcorders/3D-CAMCORDERS/model.HDC-SDT750K

[3] FinePix REAL 3D W3 | FujiFilm Global: http://www.fujifilm.com/products/3d/camera/finepix_real3dw3/

[4] J. Yang et al., "Multiview video coding based on rectified epipolar lines", International Conference on Information, Communication and Signal Processing, pp.1-5, 2009.

[5] Joint Draft 8.0 on Multiview video coding, JVT-AB204, 2008.

[6] Y. Su, A. Vetro, A. Smolic, "Common Test Conditions for Multiview Video Coding", ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Doc. JVT-T207, July 2006.

[7] X. Xu, Y. He, "Fast disparity motion estimation in MVC based on range prediction", International Conference on Image Processing, pp.2000-2003, 2008.

[8] Y. Kim, J. Kim, K. Sohn, "Fast Disparity and Motion Estimation for Multi-view Video Coding", Transaction on Circuits and Systems for Video Technology, vol.53, no.2, pp.712-719, 2007.

[9] J.P. Lin, A.C.W. Tang, "A fast direction predictor of inter frame prediction for multi-view video coding", International Symposium on Circuit and Systems, pp.2589-2592, 2009.

[10] L.-F. Ding et al., "Fast motion estimation with inter-view motion vector prediction for stereo and multiview video coding", International Conference on Acoustics, Speech and Signal Processing, pp.1373-1376, 2008.

[11] Z.-P. Deng et al., "A Fast View-Temporal Prediction Algorithm for Stereoscopic Video Coding", International Congress on Image and Signal Processing, pp.1-5, 2009.

[12] L. Shen et al., "View-Adaptive Motion Estimation and Disparity Estimation for Low Complexity Multiview Video Coding", Transaction on Circuits and Systems for Video Technology, vol.20, no.6, pp.925-930, 2010.

[13] Chang, N.Y.-C. et al., " Algorithm and Architecture of Disparity Estimation With Mini-Census Adaptive Support Weight", Transaction on Circuits and Systems for Video Technology, vol.20, no.6, pp.792-805, 2010.

[14] P.-K. Tsung et al., "Cache-based integer motion/disparity estimation for quad-HD H.264/AVC and HD multiview video coding", International Conference on Acoustics, Speech and Signal Processing, pp.2013-2016, 2009.

[15] C.-Y. Chen et al., "Level C+ data reuse scheme for motion estimation with corresponding coding orders", Transaction on Circuits and Systems for Video Technology, vol.16, no.4, pp. 553- 558, 2006.

[16] JMVC 6.0," garcon.ient.rwthaachen.de, Sep. 2009.

[17] B. Zatt et al., "Memory Hierarchy Targeting Bi-Predictive Motion Compensation for H.264/AVC Decoder", IEEE Computer Society Annual Symposium on VLSI, pp.445-446, 2007.

[18] M. Shafique, L. Bauer, J. Henkel, "3-tier dynamically adaptive power-aware motion estimator for h.264/AVC video encoding", International Symposium on Low Power Electronics and Design, pp.147-152, 2008.

[19] Y.K.-Lin et al., "A 242mW 10mm^2 1080p H.264/AVC high profile encoder chip", Design Automation Conference, pp. 78-83, 2008.

[20] G. Fukano et al., "A 65nm 1Mb SRAM Macro with Dynamic Voltage Scaling in Dual Power Supply Scheme for Low Power SoCs", NVSMW/ICMTD. pp.97-98, 2008.

[21] B. Zatt, M. Shafique, S. Bampi, J. Henkel, "An Adaptive Early Skip Mode Decision Scheme for Multiview Video Coding", Picture Coding Symposium, pp.42-45, 2010.

[22] M. Shafique, L. Bauer, J. Henkel, "enBudget: A Run-Time Adaptive Predictive Energy-Budgeting Scheme for Energy-Aware Motion Estimation in H.264/MPEG-4 AVC Video Encoder", Design, Automation, and Test in Europe Conference and Exhibition, pp.1725-1730, 2010.