A CLASS OF VECTOR-TRACING MOTION ESTIMATION ARCHITECTURES FOR MPEG2 TYPE CODING FOR TV AND HDTV

M. Gumm, F. Mombers, S. Dogimont, D. Mlynek

Ecole Polytechnique Fédérale de Lausanne, ELB-Ecublens, CH-1015 Lausanne

ABSTRACT

A class of motion estimation VLSI architectures is presented which has been developed for the use in studio quality MPEG2 encoders. A new, fast motion estimation algorithm is applied which exploits both, temporal and spatial redundancies in motion vector fields and delivers near full search quality on large search windows. The proposed architectures are MIMD based, scalable both on chip and system level, and provide high flexibility according to a programmable RISC/co-processor approach. A chip tailored to TV resolution requirements is under design. The same architecture principle can be used to build HDTV capable motion estimation devices.

1. INTRODUCTION

Digital Television based on video compression according to the MPEG-2 [1] standard is about to enter in broadcast studio environments. MPEG2 based compression techniques play also a key role in the new ATSC and DVB [2] standards for high-definition TV. Within MPEG-2 video compression, the reduction of the temporal redundancy in video sequence by motion estimation is the most computationally heavy task requiring up to several hundreds of GOPS (Giga Operations Per Second) processing power. Accurate and realistic motion estimation is crucial to both, the overall quality of the compressed video and the achievable compression rate.

The straight-forward approach to find the best match of a block of $n \times n$ pixels of the current picture in a search window of $M \times N$ pixels of a previous (or future) frame is an exhaustive search over all possible search locations, using the mean absolute error (MAE) as criterion. This method guarantees to minimize the overall PSNR. Table 1 shows some TV picture formats (the bottom row is the NTSC TV standard) and the required processing power when a full search over a relatively small search range of ±16H/±16V pixels is applied.

Picture size [pixels]	Frame rate [Frame/s]	Bit rate [Mbit/s]	Proc. power [GigaOP/s]
1920 x 1080	30 p	750	127
1280 x 720	30 p	330	57
720 x 480	60 p	250	51
720 x 480	30 p	125	26

Table 1: Comparison of ±16 pixel full search processing power for different TV resolutions (4:2.0 video signal)

To reduce this computational burden, a variety of fast search algorithms have been proposed in the past. The most commonly used approaches can be grouped into: regularly reduced search locations algorithms like the Three-Step-Search [3] or N-Step-Search, hierarchical and pixel sub-sampling based algorithms like the Multi-Grid-Search [4], or combinations of these two approaches. Full search chips for NTSC resolution video have been reported e.g. in [5], hierarchical and sub-sampling techniques are applied in [6], [7], [8]. The architectures are all based on one or several pixel processing blocks with a systolic array structure. This paper describes a class of motion estimation architectures tailored to a new high performance motion estimation algorithm for large search windows. Based on this architecture, a chip for ML@MP MPEG2 video (TV resolution) encoding is under development. Furthermore, the concept for a future HDTV capable VLSI is described.

2. ALGORITHM DESCRIPTION

The "Traced Pseudo Genetic Search Algorithm" (TPGSA) [9] combines two strategies: a genetic-based fast search strategy to reduce significantly the number of search points in a search window and a vector tracing technique to initialize this search taking into account the strong similarities of motion vector fields in adjacent frames of a video sequence.

2.1 Fast Search Strategy

The Pseudo Genetic Search algorithm (PGSA) strategy works with a strongly simplified genetic algorithm concept tailored to the needs of a realistic hardware implementation:

- Initialize the first population of motion vectors with the best motion vector of the previous macroblock in the same slice. Complete the population of 20 motion vectors by adding small, Gaussian distributed random values to the latter.
- Calculate the MAE values of all motion vectors in the first population, memorize a sorted list of the nine best vectors together with their MAE value.
- Form a new population of 20 motion vectors by the crossover operation (arithmetic mean value) and adding small, Gaussian distributed random values to them.
- Calculate for each of the 20 new motion vectors its MAE value and update the set of 9 best motion vectors if required.
- Repeat steps 3 to 4 for the third and forth generation.
- After the evaluation of the fourth generation, the first element of the memorized 9 best vectors represents the final best (integer pixel) motion vector.
- Do a half-pixel refinement of ±1 pixel around the obtained vector to form final result.

2.2 Exploiting Temporal Redundancy

In many cases, the motion of objects in a scene is approximately uniform for a time much longer than the temporal distance between two frames. I.e. the motion vector fields of consecutive frames are highly correlated. This characteristic can be exploited to significantly improve the results of algorithms like the Pseudo Genetic Search by applying a technique called *Vector Tracing* which is depicted in Figure 1. For the initialization vectors to

estimate a macroblock in P2 from P1, the best motion vector (only the frame vector) of the same macroblock position in P1 and its eight surrounding neighbors are taken. In a similar way, for the initialization vectors to estimate a macroblock in B1 or B2 from P1 or P2 respectively, the best motion vector of the same macroblock position in P2 and its eight surrounding neighbors are taken after scaling them for the appropriate position of the B-Frame in the GOP and after adjusting their sign to the required estimation direction depending on the prediction direction (forward of backward). In case that there is an I-Picture in stead of the future P picture, the best motion vectors of the past P picture are used to initialize the B-pictures search.

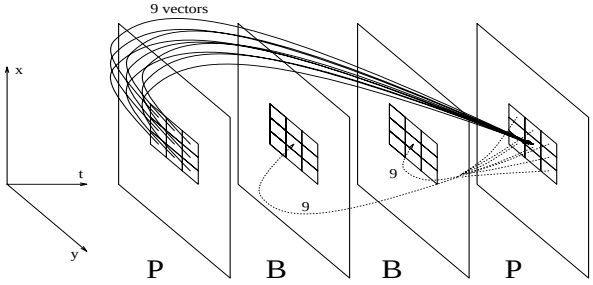

Figure 1: (from [12]) : Principle of vector tracing

2.3 Combined approach and enhancements

In combining the vector tracing technique with the PGSA algorithm we obtain the TPGSA where both, the spatial and the temporal correlation of the motion vector fields are exploited. For this, the first initial population of 20 motion vectors is formed of the nine trace vectors, the best motion vector of the previous macroblock in the same slice and 10 vectors generated by adding small, Gaussian distributed random values to the latter. TPGSA is capable to deliver nearly full-search PSNR on large search windows (±75 pixels horizontally and vertically in the simulation [9]) with only 80 (!) MAE calculations per macroblock. A side effect of vector tracing is also the smoothness of the motion vector fields.

The vector tracing approach looses effectiveness with increasing temporal distance between the consecutive P- pictures and especially when an I-picture occurs between to P-pictures. This can be simply overcome by estimating also I-pictures from past P-pictures (which is not defined in MPEG2) to generate initialization vectors for the future P-picture. For real time hardware implementation, this represents no additional computational burden as normally a motion estimator is in an idle state during an I-picture. For very complex sequences (like the basketball test sequence), the prediction quality for scenes with very fast motion can further enhanced by applying a 2-phase vector tracing scheme. In the first phase, the sequence is treated like a low delay coding sequence, and non-MPEG2 conform predictions in display order are performed with the only goal to calculate very exact tracing vectors by adding up partial results of this estimation (e.g. to form the tracing vector P→ I, the non-MPEG2 motion vectors B1→I, B2→B1, and P→B2 are added up and stored). In the second phase, MPEG2 conform motion estimation is done using the pre-calculated initialization vectors. This 2 phase motion estimation implies of course an significant increase in processing power (by 50%) due to the additional first phase.

3. IMPLICATIONS ON HARDWARE

The main characteristics of the TPGSA from the hardware point of view are:

1. non-correlated search points within the search window due to the addition of random vectors and initialization of the search with motion vectors from previous pictures
2. for one phase motion estimation: need to access best motion vectors from previous estimated pictures
3. for two phase motion estimation: need to generate (in the first phase) and access (in the second phase) the trace vectors.

The first point means that an architecture which implements the TPGSA cannot be based on the systolic array approach used for most other reported motion estimators. The complete search window must be accessible at any time, i.e. it must be stored in a cache memory on chip. The processing is dived into two steps: generation of the next candidate vector to be tested and evaluation of the MAE value of this vector. For the former, it is of advantage to use a programmable candidate vector generator, to allow for the adaptation and optimization of the algorithm to the application needs. For the latter, a fast pixel processing unit is needed which must follow a tradeoff between the latency of the calculated MAE value and the number of pixels of a macroblock which can be treated in parallel. In contrary to systolic array approaches where parallelism is exploited on pixel level, a TPGSA motion estimator can only be parallelized on macroblock level, i.e. the architecture should treat several macroblocks in parallel to satisfy given real time constraints.

The second point implicate the a TPGSA motion estimator has to store (e.g. in its frame memory) beside the video data also the found best motion vector for every macroblock (in I and P frames) and access these motion vectors for the initialization of the motion estimation of following pictures. If two-phase vector tracing shall be implemented, the processing sequence becomes even more complicated, because two motion estimation processors are required. The first on is working on pictures in display order and generates tracing vectors which are passed to the second motion estimation processor. The latter, doing the real motion estimation in MPEG2 coding order, uses these trace vectors to initialize the motion estimation for every macroblock. that means, mechanisms have to be implemented to :

- reorder incoming pictures from display to coding order
- build up successively trace vectors from frame to frame (e.g. I←B1 + B1←B2 + B2←P)
- send trace vectors from on chip to another

4. DEGREES OF FREEDOM

The following main architecture parameters can be varied to design a TPGSA motion estimator dedicated for a certain application:

- the number *N* of processing cores on one chip
- the number *P* of parallel working chips
- the number *B* of luminance pixels which can be treated in parallel by a pixel processor (= cache output width)
- the video input (=output) interface width (in bits) N_{IN}

- the memory interface width (in bits) N_{MEM}
- the ratio n_C between chip clock frequency and target video system clock

5. HARDWARE LIMITATIONS

The main factors which limit the degree of freedom for real implementations of a motion estimator are: chip area (cost), required memory bandwidth and chip IO count.

A value *B=16* has been settled as a realistic value for the pixel processor, meaning that the latency for one MAE calculation is 16 chip clock cycles with a pixel processor input data size of 16*8=128 bits. This constrains also the time for the generation of a new candidate vector which has been found to be sufficient. Once a target application has been fixed (i.e. for a TV application with 18 MHz system frequency, based on MPEG 2 ML@MP video 4:2:0 with a GOP structure of maximum 3 B pictures), the required processing power for the motion estimator can be easily calculated. The main choice to be done is to select the values for *P*, *N*, N_{MEM}, and n_C, depending on the available technology and the maximum desired area (cost) per chip. For the given TV example, when fixing N_{MEM} to 16 bit, choices can be made e.g. between (*P=1, N=4,* n_C*=3*), (*P=2, M=2,* n_C*=3*), or (*P=2, M=4,* n_C*=2*).

Once these parameters have been settled, the maximum available memory bandwidth constrains further the set of possible solutions. Today's SDRAM technology allows for very high memory bandwidths. An interface for this type of RAM is a good choice for a TPGSA processor. Even with such a fast memory interface and assuming that the internal chip memory bus is also of width N_{MEM}, the bandwidth requirements on this chip bus are very high. The "traffic" consists of: storing a new macroblock from video input to the RAM, writing a macroblock from RAM to video output, and for every processing core: loading a part of the search window and the current macroblock into the caches, sending trace motion vectors at the beginning and result motion vectors at the end of the processing of one macroblock. When these operations are summed up, it comes out that e.g. the (*P=2, M=4,* n_C*=2)* version for the example above is not realizable.

6. ARCHITECTURE DESCRIPTION

6.1 Processing Core Architecture

The processing core (Figure 2) of the proposed TPGSA motion estimator consists of:

- the search window caches which must be large enough that for TV and HDTV applications the coding quality can be kept very high. For TV, a size of 96*64 pixels has been simulated with very good results. Other sizes are possible, constrained by area and memory bandwidth limits.
- the cache for the current macroblock
- the pixel processor which can treat 8 luminance pixels in parallel, calculates three MAE values on the fly (1 16*16 line and two 16*8 line values), with optional half pixel and dual prime capabilities (described in more detail in [11]).
- the 24-bit vector generation RISC controller, an ASIP controller with sub-word parallelism (to treat the 2 12-bit motion vector components in one cycle) with its own instruction memory of 1024 instructions (described in more detail in [12]).
- a random number generator with pseudo Gaussian distribution
- a unit which sorts the motion vector/MAE pairs and interfaces to the chips memory bus.

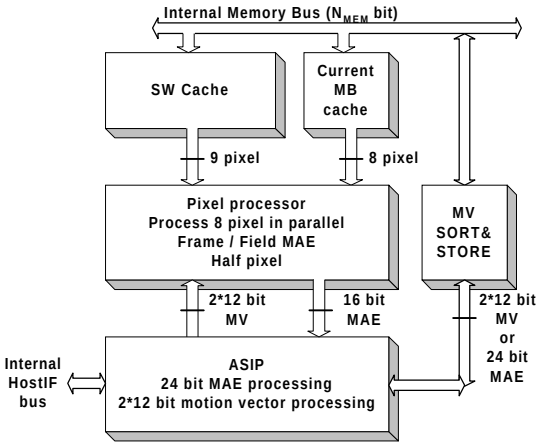

Figure 2 : Architecture of one processing core

6.2 Chip Architecture

The chip architecture is depicted in Figure 3.

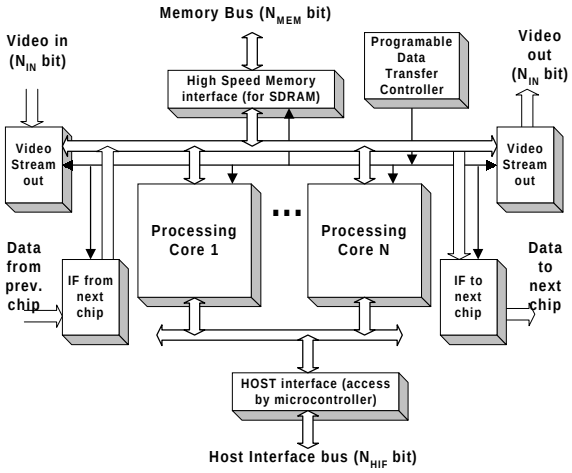

Figure 3 : Proposed architecture model. The parameters *N* of processing cores and the bus width N_{IN} and N_{MEM} can be chosen to fit the envisaged application.

N processing cores are working in parallel (MIMD type architecture). A programmable data transfer controller handles all the different data transfers on the internal memory bus. Video input and output interfaces must be adapted to the target video systems protocols and needs or can be implemented with a general protocol. The high speed memory interface is able to control SDRAM-like memory devices to guarantee high throughput. Through the host interface, the architecture can be configured (at processes begin or at picture rate). This block can also be tailored to the video target system needs or be kept general to interface with a range of DSPs and micro-controllers.

Special interfaces have been included into the architecture to allow for easy scalability of several chip by transferring temporal results from one chip to another during the processing. Scalability can be use e.g. to enlarge the search window to 4 times the size covered by only one chip.

7. THE "IMAGE[1]"CHIP

Based on the described architecture concept, a chip is currently under design which implements a motion estimator for TV resolution MPEG2 ML@MP 4:2:0 video processing. This work is done in frame of the European ATLANTIC project [10]. The chip is called IMAGE (Integrated MIMD Architecture for genetic Motion Estimation) and integrates two processing cores, a fast 16 bit SDRAM interface, ±40H/±24V search range per processing core and dedicated interfacing circuitry for the target video system. 1 chip is sufficient for low-delay coding applications, 2 chips for coding B pictures, 3 chips are needed to form a 2-phase vector tracing system, and a set of 2*4 chips enlarges the search range to ±80H/±48V pixels (for B picture coding). the main chip data is given in Table 2.

IMAGE processing power	
Pixel Processing power	3.5 GOPS / 108 MIPS
Picture format	720x576@25 fps; (4:2:0)
Memory interface	16 bit / 54 MHz
Number matchings / MB	120
Operating frequency	54 MHz
Implementation Technology	4 layer 0.35 um CMOS
Estimated chip area	40 mm^2
Transistor count (pre-layout	1.5 M with memory
On-chip RAM	~25 Kbyte

Table 2 : IMAGE chip data

8. A HDTV MOTION ESTIMATOR

In this section, a realizable architecture for mid-resolution, HDTV application is proposed, derived from the described architecture family. Table 3 summarizes the features and estimated data of this HDTV chip. Estimates for transistor count and area are based on the real design and technology data from the IMAGE chip.

HDTV motion estimator	
# parallel processing cores	4
picture format	1280x720@30 fps; (4:2:2)
search window size	±64H/ ±40V pixels
operating frequency	72 MHz
Implementation Technology	4 layer 0.35 um CMOS
Estimated chip area	100 mm^2
Estimated transistor count	3 M with memory
Memory interface	32 bit / 72 MHz

Table 3 : features of the proposed HDTV motion estimator

[1] this design is done in cooperation with the CSELT research laboratory in Torino/Italy. To be published soon.

Coming up from this design, the task of designing the HDTV chip could be accomplished in a short time because of the high regularity of the proposed architecture family and the reusability of IMAGE design parts, mostly done using VHDL and synthesis.

9. SUMMARY

A new class of motion estimation architectures has been described which are designed to execute the new, high performance TPGSA search algorithm. The architectures are based on parallel working processing cores, each equipped with its own ASIP, pixel processor and search window cache memory. The processing cores are supplied with the required data via an high speed memory interface controlled by a programmable chip controller. The number of processing cores, the memory interface width, and several other parameters can be varied for a chip solution tailored for its application, from low cost encoders to HDTV mid resolution equipment. An architecture for TV studio application is under design, another architecture for HDTV applications has been proposed.

10. REFERENCES

[1] ISO/IEC 13818-2:1996 Information technology - Generic coding of moving pictures and associated audio information: Video

[2] DVB standard: http://www.dvb.org

[3] J. Koga, K. Iiunuma et al., Motion Compensated Interframe Coding for Video, Proc. of the national telecommunications Conference, 1981

[4] F. Dufeaux, M. Kunt, Multigrid Block Matching Motion Estimation With An Adaptive Local Mesh Refinement, Visual Communications and Image processing'92, Nov. 1992

[5] K. Ishihara et al., *A Half-Pel Precision MPEG-2 Motion Estimation Processor with Concurrent Three Vector Search*, Digest of techn. papers, IEEE Intrn. Solid State Circuits Conf, pp.288-289, February 1995

[6] K. Suguri et al., *A Real-time Motion Estimation and Compensation LSI with Wide Search Range for MPEG2 Video Encoding*, IEEE Journal of Solid States circuits, Vol. 31, No. 11, pp.1733-1740, November 1996

[7] M. Mizuno, Y. Ooi et al., *A 1.5 W Single-Chip MPEG2 MP@ML Encoder with Low-Power Motion Estimation and Clocking*, Digest of techn. papers, IEEE Intrn. Solid State Circuits Conf., pp. 256 , February 1997

[8] R. Pacalet, A. Lafage et al. (ENST, Philips), *A Real Time MPEG2 MP@ML Motion-Estimator Chipset*, Digest of techn. papers, IEEE Intrn. Solid State Circuits Conf., pp. 260, February 1997

[9] M. Mattavelli, D. Nicoulaz: *A Low Complexity Motion Estimation Algorithm for MPEG-2 Encoding*, Proc. of the Hamlet RACE2110 workshop, February 27-28th, 1996, Rennes

[10] ATLANTIC: http://www.bbc.co.uk/atlantic/

[11] F. Mombers, M. Gumm et al., *A video Signal Processing core For motion Estimation in MPEG2 Coding*, Proc. of ISCAS, Vol4., pp 2805-2808, Hongkong, 1997

[12] S. Dogimont, M. Gumm et al., *Conception and Design of A RISC Controller for the Use as Embedded Controller within a Parallel Multimedia Architecture*, Proc. of ASAP, pp.412-421, Zürich, 1997