COMPLEXITY CONSTRAINED MOTION ESTIMATION ALGORITHMS FOR MPEG2 VIDEO ENCODING

M. Mattavelli §, F. Puggioni §, C. Lopez §F. Bellifemine ‡

§Signal Processing Laboratory Swiss Federal Institute of Technology, CH-1015 Lausanne, Switzerland.
‡CSELT, Via G. Reiss Romoli, 274 – Torino, Italy.

RÉSUMÉ

Cet article aborde l'étude de divers algorithmes d'estimation de mouvement dans un but d'optimisation de codage selon la norme MPEG2. L'analyse est faite sur la base du compromis entre performance et complexité tout en adaptant les algorithmes à une architecture de circuit donnée. La complexité des algorithmes "full search", multi grille, N-step et génétique a été considérée. Plus particulièrement, un algorithme génétique a été développé et adapté de sorte à optimiser ses performances dans un contexte d'estimation de mouvement par appariement de blocs. De plus, diverses méthodes faisant usage de la redondance spatiale et temporelle du champ de vecteurs de mouvement de la structure de groupe d'images de MPEG2 ont été introduits. Les modifications proposées produisent un algorithme de faible complexité d'implantation matérielle dont les performances de codage sont supérieures aux autres algorithmes génétiques décrits dans la littérature.

ABSTRACT

This paper studies the performances of some motion estimation algorithms for the optimization of MPEG2 video encoding. The analysis is based on the evaluation of the coding performance versus the complexity, constraining the considered algorithms by all the limitations of a real hardware architecture. Full search, multigrid search, N-step search and genetic search have been considered for the complexity constrained evaluation. In particular a genetic motion search *algorithm has been developed and adapted in order to optimize its performances for the block motion estimation problem. Moreover further improvements that exploit temporal and spatial uniformity of motion vectors in the group of picture structure of MPEG2 have been introduced. The proposed modifications yield an algorithm which requires low-complexity hardware implementations and whose coding performances are higher than other genetic algorithms proposed in literature.*

1 Introduction

Motion estimation (ME) is an important problem for the optimization of MPEG2 video encoding. The temporal redundancy present in the video sequence is exploited for compression by compensating the motion between successive pictures: the better the motion estimation, the higher the efficiency of the video encoding process. However, the MPEG2 standard defines only a decoding syntax or, in other words, the coding structures that can be used for generating the encoded bitstream. The usage of motion vectors is specified for each prediction mode but nothing in the standard specifies how to find these vectors for the achievement of the best video quality. Not only ME is relevant for MPEG2 encoding performances, but it is also the most demanding process in terms of complexity and memory access bandwidth.

This work addresses the problem of the coding performance versus the complexity in two steps. The first consists in considering different ME algorithms and optimizing their performances specifically for MPEG2 encoding. The second consists in introducing further optimization techniques that take advantage of the temporal and spatial statistical distribution of motion vectors in the group of picture (GOP) structure of the MPEG2 standard. Performances have been evaluated by encoding some test sequences with the MPEG-2 TM5 [1] encoding algorithm at 9 Mbit/s. The target applications of MPEG2 will require, at least for the near future, dedicated hardware architectures. Therefore the basis of the comparison for the different algorithms and optimization issues is a common hardware complexity and not a generic algorithmic complexity.

The paper is organized as follows: next section discusses the problem of the mapping of motion estimation algorithms into an hardware architecture and presents the criteria used to define common complexity levels. Section 3 briefly presents the algorithms that have been considered for the optimization and comparison. In section 4 the main modifications applied to the *genetic* algorithm in order to make it suitable for block ME are described. Optimization techniques that exploit spatial and temporal motion uniformity and their use for MPEG2 ME are discussed and described in section 5. Section 6 reports the MPEG2 coding results of the studied algorithms and optimization techniques. Section 7 finally presents the conclusions.

2 Motion estimation algorithms and hardware architectures

In order to find the best coding performance versus complexity for MPEG2 motion estimation it is necessary to define a complexity for the studied algorithms. The simple computation of the number of operations or the simulation time, do not correspond directly to the hardware complexity necessary to implement the considered algorithms. In fact algorithms that presents a higher regularity can be implemented more efficiently in appropriate architectures than apparently simpler ones that are characterized by non-regular behaviors. In this work all algorithms have been mapped into a common hardware architecture used as reference. This architecture is the *block matching processor* described in [2]; it is programmable and can implement with high efficiency different block motion estimation algorithms. In this way it has been possible to consider all the constraints of real architectures including memory access bandwidth limitations. These limitations are usually neglected in most works for the comparisons among different algorithms, but they constitute a strong hardware constraint when relatively large search areas are used by the estimation algorithms. In general a common complexity level can be set considering a maximum number of block matchings and limiting the freedom of the displacements among successive matchings. In this way performances of algorithms that require sparse or irregular displacements and that do not respect the constraints, are evaluated with a lower number of matchings in order to remain in the allowed memory access bandwidth.

3 Selection of the algorithm

Four algorithms have been considered for the study: *full-search, n-step search, multigrid search* and *genetic search.*

Full search algorithm (FSA) searches for the vector that yields the lowest matching error of an *a-priori* defined search area, known as *search window.* It always finds the global minimum of the considered area. In the case of sequences characterized by small displacements, the algorithm can be quite efficient because the true movements fall in the considered *search window.* Instead when large movements are present in the video sequence it is necessary to use a large *search window* and a large number of block matching computations. When FSA is compared with other algorithms at the same level of complexity it is not guaranteed to produce the best encoding performance.

The *N-step search* algorithm (NSSA) is based on the iterative vector search on points belonging to a coarse grid created around the vector corresponding to the best match of previous iteration [3]. At each step the grid becomes finer and the algorithm stops when the vectors are searched on a grid with the desired resolution. Many combinations of grid resolutions and search window sizes are allowed. Much larger search windows are also possible, but since the algorithm is based on the assumption, usually not correct, that the matching error decreases monotonically as the searched point moves closer to the global minimum, its results might not be reliable because the algorithm can be easily trapped in local minima. No limitations are imposed by the memory access bandwidth since the search is performed on regular grids with the same horizontal or vertical coordinates. Single input-output operations can be used for multiple block matching evaluations.

The third algorithm considered in this work is the *multigrid-search algorithm* (MSA) [4]. It is based on the principle of using different matching window sizes to find and successively refine robust estimate of the displacements vectors. First large displacements are detected by matching large blocks on the points of a coarse grid; in this way an initial robust estimation of the displacements is obtained. In a second step the first vector estimates are refined on a finer grid with smaller matching blocks and smaller search windows. Many combinations of block size, search windows and fine-to-coarse and coarse-to-fine strategies are possible. This algorithm is more reliable than the NSSA. In fact, appropriate strategies of coarse-to-fine and fine-to-coarse grid search can avoid being trapped in local minima. Unfortunately the number of matching necessary for implementing these strategies limits the potential performance of the algorithm. Thanks to the regularity of the search grids no other limitation needs to be imposed by the memory access bandwidth.

The last algorithm considered in this work is the *genetic-search algorithm* (GSA) [5]. It is based on the natural selection principle characterized by *the survival of the fittest.* A set of potential solutions to the considered problem is processed. This set called *population* belongs to a given search space. The algorithm operates on coded representations of the solutions. Each coded solution is called *chromosome*, in analogy with nature, and each *chromosome* is composed by some *genes* that represent the genetic materials of the element of the *population.* The algorithm evolves by exchanging *genes* between *parents* for creating the elements of the new *generation.* In more details the steps characterizing a GSA are the following:

1. an initial *population* is created in accordance with a certain probability distribution,

2. a *fitness* value is associated to each element: the higher the value, the higher the probability of survival of the element,

3. selected among the best *fitness* values, pairs of population elements are chosen with a given probability in order to generate the elements of the new *population.* The generation is done by a operation called *crossover* that allows the interchange and combination of the *genes* of *parents chromosomes* for obtaining the *sons*,

4. since it may happen that the algorithm converges to a point where all the *chromosomes* present uniform characteristics, another operation called *mutation*, that occurs with a given probability, allows a random change of the *genes*; in this way new possibilities are given to the algorithm of finding different good *chromosomes*,

5. these last three steps are repeated for each *generation* until a convergence criterion previously established is reached. The criterion can be the convergence of the solution, or a maximum number of *generations*. In the case of ME it has been set to a given product of population size and number of *generations* corresponding to the chosen complexity level.

A number of simulations have been performed on different test sequences and different levels of complexity with the MPEG2-TM5 encoding algorithm at 9 and 4 Mbit/sec. NSSA and MSA in all cases show systematically poorer performance than FSA and GSA. In the case of NSSA many different strategies concerning grid resolutions and sizes have been tested without relevant improvements. The results obtained confirm the criticism mentioned above about the drawbacks of the method. The MSA instead results penalized since, when compared at the same level of complexity with the other algorithms, it does not allow larger window search sizes. The advantages of the MSA approach, although relevant in other contexts [4], do not yield significant gains for MPEG2 encoding.

GSA instead shows performance globally superior to all the other tested algorithms. But the implementation of the algorithm in its *classical* form [6] does not seem the best choice for the ME problem. An accurate analysis of its behavior has shown some problems and inefficiencies that require appropriate modifications of the algorithm. These modifications yield relevant improvements of the coding performance. Moreover the different parameters that define the various operations need an appropriate setting to make the most of the GSA approach.

4 A modified genetic search algorithm

The first question to be answered is the choice of the most efficient mapping of the vectors into the *chromosomes* and of the *crossover* operation. We have found that the interleaved binary representation and binary bitwise *crossover* proposed in [6] is not the best choice for ME. The best results have been found assuming a vectorial mapping of the vectors into the *chromosomes* and defining the *crossover* as the half norm vectorial sum of the *chromosomes*.

Another problem is the definition of the best initial population distribution. A good random distribution has been found to be Laplacian with a variance given by the size of the defined search window. If a vector falls outside the window it is discarded and a new vector is generated randomly.

The *mutation* operation has been tested in various options regarding its probability occurrence and its vector representation. Only negligible or small decreases of the performance have been observed, hence this stage has been completely eliminated.

Due to the small dimensionality of the search space (only two dimensions and two *genes*) the resulting genetic material may somehow be limited in its *genetic variety*. In fact it has been verified experimentally that it tends to generate several time the same vectors. Since the block matching process is by far the most expensive operation, the algorithm has been modified in order not to execute matching for repeated entry. New vectors are generated for the evaluation but the repeated vectors are kept in the population since their genetic material should not be lost for the next generations.

Another problem found in the experiments is the loss of the optimal vectors. In fact in GSA only the population elements that are present in the last *generation* are chosen as an optimal solution. If the algorithm is stopped by a criterion that is not the convergence one, the optimal solution found in a previous *generation* can be lost. The choice of the optimal vector is done among the best for each *generation*.

5 ME optimization techniques for MPEG2 group of pictures

The GSA, in opposition to the other algorithms considered for the comparison, is characterized by many options and parameters that define its behavior. Hence, for its flexibility, it is possible to include in a natural way other optimization options linked to the specific ME problem. One key factor for the performance of the GSA is the initialization of the first population. A random distribution is usually the best choice for the setting of the population elements.

In video sequences the movements of the elements in the scene are characterized by a degree of correlation in space and time. The GSA can exploit these correlations without any increase in the complexity of the algorithm by introducing in the initial population *good* candidates. Hence in addition to the random vectors, some other vectors that correspond to the hypothesis of space uniformity of motion (neighbors vectors) and time uniformity of motion (tracking of the motion trajectory) are introduced in the first generation. In our implementation we have chosen to use only the vectors that are actually used by the MPEG2 motion compensated prediction. This means tracking the motion following the coding order of the GOP. This option does not increase the complexity of the algorithm and requires only the additional memory for the motion vectors of one frame. Other options like for instance tracking the motion following the display order, would require instead the additional estimation of these tracking vectors that cannot be used for the actual motion prediction. Figure 1 reports the graphical representation and the dependencies in the MPEG2 GOP of the vectors used for the initialization of the first population.

6 Simulation Results

Simulation results of the sequences *Flower Garden* and *Basket Ball* encoded at 9 Mbit/sec, frame prediction with the TM5 algorithm are reported in Figs. 2 and 3. The curves represent the average PSNR gain calculated over the first 8 GOP ($M = 3$ $N = 12$, 96 frames) in function of the size of the search window. Results corresponding to the modified GSA with and without the optimization features developed for the MPEG2 GOP structure are reported. The 0 dB level

corresponds to the performance of the FSA on a window size of 9 pixel which is equivalent to the complexity of the GSA. The GSA yields gains in the range of 2-3 dB. It approaches the results achievable by a FSA extended over the window search size (dashed line), which can be considered as a theoretic limit and requires a complexity ranging from one up to two order of magnitude higher.

7 Conclusion

Four block ME algorithms, *full search, n-step search, multigrid search* and *genetic search* have been studied for the optimization of MPEG2 encoding. The comparison among the different algorithms is based on a performance versus complexity evaluation mapping all the algorithms in a real hardware architecture. The GSA in its *classical* form, although showing the best performance among the considered algorithms, has been modified in order to further improve its performance for the specific ME problem. Moreover, optimization techniques that exploit spatial and temporal uniformity of motion vectors in the group of picture structure of MPEG2 have been included in the GSA. The optimized algorithm yields relevant coding gains with no increase of the complexity compared to the other ME algorithms.

Acknowledgments The authors whish to thank the partners of RACE 2122 COUGAR project for the permission of publishing this work and Dr. F. Dufaux for the kind collaboration.

References

[1] International organisation for standardisation. Coded representation of picture and audio information: Test model 5. Technical report, ISO-IEC/JTC1/WG11.

[2] J. Kowaltczuk, R. Hervigo, and D. Mlynek. A multiprocessors architecture for an hdtv motion estimation system. *IEEE Trans. on Consumer Electronics*, Vol. 38, No. 3, pp. 690–697, August 1992.

[3] J. R. Jain and A. K. Jain. Displacement measurement and its application in intraframe image coding. *IEEE Trans. on Communications*, Vol. COM, No. 29, pp. 1799–1808, Dec 1981.

[4] F. Dufaux. Multigrid block matching motion estimation for generic video coding. *Ecole Polytechnique Federale de Lausanne, These 1221*, 1994.

[5] M. Srinivas and L.M. Patnaik. Genetic algorithms: a survey. *Computer-IEEE*, pages 17–26, June 1994.

[6] K. H.-K. Chow and M. L. Liou. Genetic motion search algorithm for video compression. *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 3, No. 6, pp. 440–445, Dec. 1993.

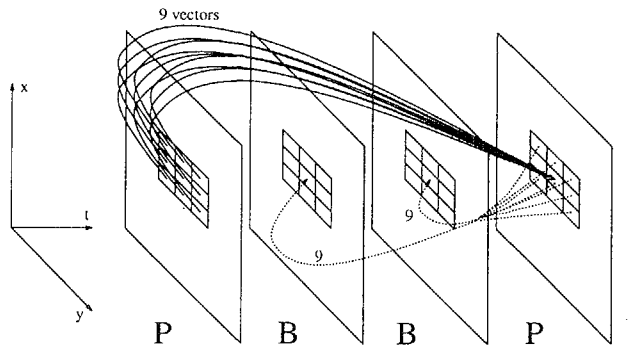

Figure 1: Graphical representation of the vectors used for the initialization of the first generation of GSA in a MPEG2 GOP.

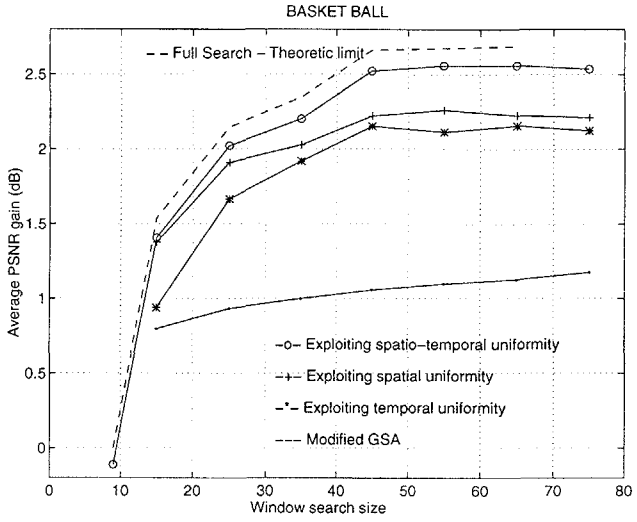

Figure 2: Luminance average PSNR gain of GSA for the sequence *Basket* versus the search window size and different optimization options.

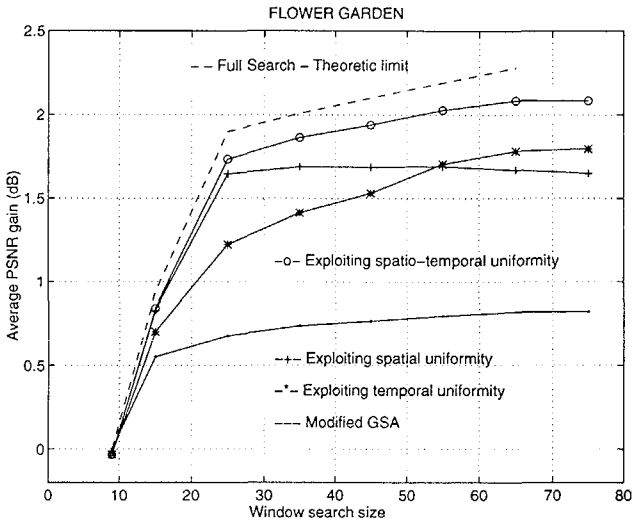

Figure 3: Luminance average PSNR gain of GSA for the sequence *Flower Garden* versus the search window size and different optimization options.