

# A PARALLEL BLOCK MATCHING TECHNIQUE FOR MPEG-2 MOTION ESTIMATION

Honorius Gâlmeanu  
Dept. of Electronics and Computers,  
Transylvania University of Braşov,  
Politehnicii 1-3,  
2200 Braşov, ROMÂNIA  
e-mail: galmeanu@vega.unitbv.ro

Daniel Grosu  
Div. of Computer Science  
The University of Texas at San Antonio  
6900 North Loop 1604 West,  
San Antonio, TX 78249-0677, USA  
e-mail: dgrosu@cs.utsa.edu

**Abstract:** *Motion estimation is the most computing-intensive component of the MPEG-2 encoding process. It refers to the action of searching for the closest prediction of a certain image block. The more accurate the prediction, the better achieved compression and quality are. The optimal solution is guaranteed by performing an exhaustive search for all possible predictions for a certain block. The motion estimation component is characterized by a polynomial time, and it is responsible for about 90% of the encoding time. In this paper the search time is reduced by parallelizing the block matching search algorithm. Experimental results show that significant improvements in performance can be made. The encoding time is reduced from 14 sec/frame, in the sequential case, to 3.2 sec/frame, about 5 times faster, using 6 Pentium 450 Mhz in parallel.*

**Key words:** - MPEG, motion estimation, motion vector, block matching, PVM.

## 1. INTRODUCTION

Motion estimation refers to the way of constructing prediction for a certain block in an image, called *macroblock*. A video sequence consists of successive static frames whose encoding becomes a target MPEG bitstream [5] to be sent over a computer network at a specified bitrate. Each frame is a dense grid of adjacent pixel macroblocks. Depending on the coding pattern, each frame can be: *intra coded*(I), encoded independently of other frames; *predicted*(P), encoded as differences from previous frame; or *bidirectionally predicted*(B), encoded as differences from both previous and next frame. The differences are obtained via motion estimation algorithms.

The main goal of motion estimation algorithms is the accurate determination of a prediction macroblock for which the absolute difference from the current macroblock is minimum. Such a macroblock roughly approximates the true motion that appears in a video sequence. The prediction macroblock is completely determined by the image it is part of, the displacement between it and the macroblock for which this prediction is determined and the corresponding prediction error. The displacement is referred with the term of *motion vector*, and it is the variable to be determined by motion estimation algorithms. The possible values for a motion vector determines the characteristic *search window*. A macroblock is coded by storing in the bitstream only the estimated motion vector and attached prediction error, efficiently coded with discrete cosine transform [1]. The MPEG standard does not impose nor recommends any motion estimation algorithms for determining the motion vectors, although the proper choice of these ones greatly impacts both on compression performance and image quality.

Parallel MPEG video encoding tries to minimize the time spent on frame encoding. A strategy for parallelizing the block matching motion estimation for MPEG-4 video sequences as described in [6]. A scalable MPEG-2 parallel encoder is presented in [2].

In this paper we present a simple approach to parallelize the block matching motion estimation algorithm for message-passing systems. This paper is organized as follows. Section 2 describes our parallel algorithm and the parallel encoder model. Section 3 summarizes the experimental results. Finally Section 4 presents the goods and the bads of our approach and some future works.

## 2. PARALLELIZATION OF BLOCK MATCHING

The classic block matching algorithm [4] requires a full search to find the best candidate for each macroblock in a frame. Our parallelization technique can be formalized as:

```

for all frames do
  for 1 to  $NPROC$  do in parallel
    for 1 to  $no\_mblocks/NPROC$  do
      find motion vector
  
```

The time required for encoding a frame is  $O(N^4)$ , where  $N$  is the frame dimension. We reduced this time by evenly distributing macroblocks in contiguous slices among the processors. Each one of the available  $NPROC$  processors (here called *estimator*) performs an exhaustive search among all macroblocks in a defined search window for finding the best prediction of each macroblock of all  $no\_mblocks$  macroblocks of a frame in the slice it received. The slice for a processor contains  $no\_mblocks/NPROC$  macroblocks. The parallel encoder model is depicted in Figure 1.

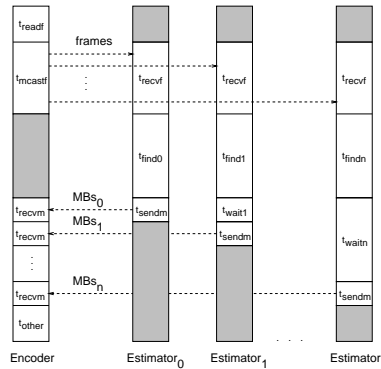


Figure 1: Master-slave model of the encoder.

A cycle time  $t_{cycle}$  for encoding one frame is contributed by the following:

- the encoder reads current frame in time  $t_{readf}$  then sends it to estimators in time  $t_{mcastf} = t_{recvf}$ ;
- each estimator determines the solutions for all macroblocks from the slice of macroblocks it received by the time  $t_{find0} \dots t_{findn}$  since it finishes receiving frames;
- the encoder eventually collects results (i.e. motion vectors and prediction errors) for all macroblocks of the frame, from every estimator, in time  $t_{recvm} = t_{sendm}$ ;
- idle times  $t_{wait1} \dots t_{waitn}$  exist due to the sequential way the master encoder receives results from slave estimators.

According to the master-slave model, there are two components which determines the total encoding cycle time, where  $t_{cycle} = t_{comp} + t_{comm}$ :

$$t_{comp} = \max(t_{find0} \dots t_{findn}) + t_{readf} + t_{other} \quad (1)$$

Srch wind.	Seq. srch	Parallel srch. (no. of estimators)							
		2	3	4	5	6	7	8	9
8 × 8	0.62	0.56	0.53	0.56	0.59	0.62	0.68	0.76	0.79
16 × 16	1.12	0.76	0.71	0.68	0.68	0.71	0.76	0.82	0.88
32 × 32	2.47	1.44	1.21	1.06	1.00	0.97	1.03	1.06	1.12
64 × 64	5.97	3.15	2.53	2.06	1.85	1.68	1.62	1.59	1.71
128 × 128	14.09	7.09	5.74	4.41	3.79	3.29	3.09	3.00	2.94

Table 1: Average time (in seconds) for encoding a frame.

$$t_{comm} = t_{mcastf} + n \cdot t_{recvf} \quad (2)$$

The variation of search window's size  $S$  and that of the number  $n$  of estimators has a great impact on  $t_{comp}$  and  $t_{comm}$ :

- an increase of  $n$  determines reduction of  $t_{find}$ , in fact  $t_{comp}$ , but increases  $t_{comm}$ ;
- enlarging the search window, both  $t_{comp}$  and  $t_{comm}$  increase in magnitude.

The main goal is to achieve as low  $t_{cycle}$  as possible, meaning that for a particular search window an optimum number  $n$  of estimators must be determined.

### 3. EXPERIMENTAL RESULTS

The parallel MPEG-2 software encoder was tested on nine Pentium II 450MHz machines running PVM 3.4.1 [3] under Linux 2.0.22, connected through an Ethernet 10BaseT hub. A sequence of 34 frames representing a motion video scene at a resolution of  $352 \times 240$  at 25 frames/sec was encoded. The original video throughput of 50.688Mbps was compressed at a 500Kbps bitrate without a visible degradation of original image quality. Table 1 shows the average time per frame for encoding this video sequence using different search windows and different number of processors. In all the experiments the macroblock's size is 8 by 8 pixels.

The time taken to compress a frame of the video sequence, using a 128x128 search window, for the sequential approach was 14 sec, while the parallel approach, using 6 estimators working independently, required only 3.2 sec, about five times faster, and preserving the same image quality.

In order to compare the performances of the parallel implementation for different sizes of the search window with those of the sequential implementation, the speedup was determined for each case. Speedup determines how many times a parallel implementation runs faster than its sequential correspondent. Figure 2 shows the speedup for various search windows. It can be seen from the shapes of the curves that there is a limitation on the number of processors that can be used efficiently. For a 64x64 search window the maximum speedup is obtained using 8 processors. Adding more than 8 processors the speedup decreases. This limitation is due to the high cost of communication in a network of workstations and it represents the major drawback of such systems.

### 4. CONCLUSIONS AND FUTURE WORKS

This paper has proposed a simple approach to parallelizing MPEG motion estimation using message-passing systems. To test the performance of our approach a parallel MPEG-2 encoder was developed. It was able to encode a frame of a test sequence in 3.2 sec using

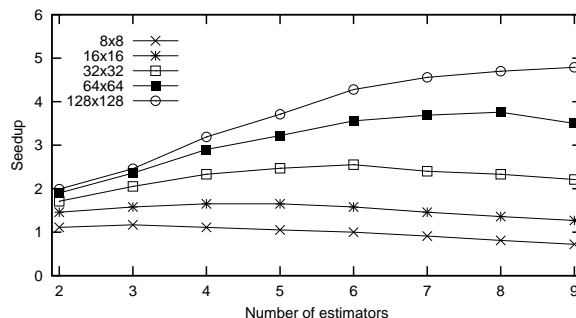


Figure 2: Variation of speedup with the size of the search window and number of estimators.

6 Pentium II 450 Mhz PCs, about 5 times faster than the time obtained using only one PC.

The speedup was studied to determine the optimal number of processors with respect to the search window's size. For a search window of size 64x64 we determined an optimum number of 8 processors.

The results are promising and we believe that further improvements are possible. In our future work, we will explore the possibility of overlapping communication and computation to improve the encoding time. Also we will modify our approach to be suitable for execution on a heterogeneous network of workstations.

**ACKNOWLEDGEMENT:** This work was supported entirely by Deuromedia GmbH. as part of the collaboration contract agreed with Transylvania University of Braşov.

## REFERENCES

- [1] N. Ahmed, T. Natarajan and K.R. Rao, Discrete Cosine Transform, *IEEE Trans. Comput.*, vol. C-23, pp. 90-93, Jan. 1974.
- [2] S.M. Akramullah, I. Ahmad and M.L. Liou, Performance of Software-Based MPEG-2 Video Encoder on Parallel and Distributed Systems, *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 7, No. 4, August 1997.
- [3] A.L. Beguelin, J.J. Dongarra, G.A. Geist, W.C. Jiang, R.J. Manchek, B.K. Moore, V.S. Sunderam, *PVM: Parallel Virtual Machine. A User's Guide and Tutorial for Networked Parallel Computing*, MIT Press, Cambridge, Massachusetts, 1994.
- [4] F. Dufaux, F. Moscheni, Motion Estimation Techniques for Digital TV: A Review and a New Contribution, *IEEE Proc.*, Vol. 83, No. 6, June 1995.
- [5] ISO/IEC 13818-2, Information technology - Generic coding of moving pictures and associated audio - Part 2: Video, 1995.
- [6] M.K. Steliaros, G.R. Martin and R.A. Packwood, Parallelization of Block Matching Motion Estimation Algorithms, Research Rep. RR-320, Department of Computer Science, University of Warwick, Coventry, UK, January 1997.