PARALLELIZED ROBUST MULTIRESOLUTION MOTION ESTIMATION

P. Piscaglia, B. Macq

Université Catholique de Louvain
Louvain-la-Neuve
Belgium
Email : piscaglia@tele.ucl.ac.be

E. Mémin, P. Pérez, C. Labit

IRISA/Centre INRIA-Rennes
RENNES
France
Email : memin@irisa.fr

ABSTRACT

Motion estimation is crucial in domains such as robot vision or image sequence coding. Fast or real time computing is often required but is problematic due to computational complexity. In this paper, an algorithm using multiresolution multigrid Markov random fields will be paralellized. Such realizations have already been studied for fine parallelizations on massively parallel machines. The goal of this paper is to achieve a coarse parallelization on a network of workstations, using PVM.

1. INTRODUCTION

Many techniques have already been proposed for motion analysis of image sequences [6, 8]. This paper will focus interest on global energy function based motion estimation algorithm. The general issue is to find the global minimum of an objective (also called energy) function describing the interaction between *observation variables* which correspond to the representation of the observed data and *hidden variables* which are the information to be extracted from the original images. Standard regularization approaches as Markov Random Field (MRF)-based image analysis [2] lead to the minimization of such global energy functions. *Multigrid* techniques [3] have been introduced with the purpose of boosting the speed of relaxation algorithms involved in the estimation process. *Multiresolution* strategies have been also widely considered in motion estimation applications in order to deal with long range displacements [4]. These two kinds of frameworks are very often mixed up in the image analysis community.

Despite the use of those multigrid and multiresolution methods, computation time remains very high. Parallelization techniques for these algorithms have been studied for a long time. Stochastic or deterministic relaxation algorithms involved in the global optimization of such problems are *regular* and *local*, leading naturally to massive parallelism. Therefore, SIMD massively parallel machine and "fine grain" parallelization have been mostly considered for these algorithms. Such machines have the asset to be fast and efficient for this set of problems, but present the drawback to be quite expensive [7, 9].

We will try in this paper to make use of usual workstations network to reduce the estimation time of motion vectors. Such networks have the important advantage to be very cheap compared with massively parallel machines. Indeed, nearly every lab or plant possesses a network of workstations, usually underused. Gathering them into a parallel virtual machine gives a nearly free important computing power. They have not the same transmission characteristics as dedicated parallel machines have, and we will have to deal with such slow communications. They normally perform very well with coarse parallelization, involving a few communications and heavy computations. The aim of this paper will be to combine and adapt characteristics of the algorithm and the network in order to succeed in an efficient implementation on the workstations, using the parallel PVM library.

2. THE PARALLEL VIRTUAL MACHINE

PVM [1] stands for Parallel Virtual Machine. It is a software package that allows an heterogeneous network of parallel and serial computers to appear as a single large distributed memory parallel computer : the *Virtual Machine*. PVM supplies the functions to automatically start up *tasks* (a unit of computation in PVM, analogous to a Unix process) on the virtual machine and allows them to communicate and synchronize with each other. By sending and receiving messages, multiple tasks of an application can cooperate to solve a problem in parallel. Usually, a master task initi-

ates slave tasks, performing computations with data received from their master, and sending back results.

3. MOTION ESTIMATION ALGORITHM

The framework of the robust multiresolution motion estimation algorithm is the estimation of motion or velocity vectors between two consecutive images $f(t)$ and $f(t+dt)$. Let $w = \{w_s, s \in S\}$ be the unknown velocity field, defined on the rectangular pixel lattice S. Let define $\nabla f = [f_x, f_y]^T$ and f_t be respectively the spatial gradient and the temporal partial derivative of the luminance.

Assuming that the luminance of a given "physical point" does not change much between times t and $t+dt$, and using as estimation criterion the *Maximum A Posteriori*, the most probable realization of the w field will be given by :

$\hat{w} = \arg\min_{w \in \Omega} H(w; f)$, with $H(w; f) \triangleq$
$\sum_{s \in S} [\nabla f(s,t).w_s + f_t(s,t)]^2 + \alpha \sum_{<s,r> \in C} \| w_s - w_r \|^2$

This estimator is classically composed of two terms. The first component of the energy, *the data model*, expresses (pointwise) interaction between the hidden labels to estimate and the observations. The second ones, called the *prior model*, encodes constraints on the desired solution.

To allow the recovery of large displacements, a standard technique consists in using a multiresolution model along with a *coarse-to-fine* estimation [4]. To this end pyramids of images are derived from the original frames by (Gaussian) smoothing/subsampling. The created pyramidal structure is then used to estimate incrementally the velocity field (see figure 1.a).

The introduction of robust estimators in the energy formulation at each resolution permits to cope with large deviations (called *outliers*) to the data model (data outliers δ_s) or from the prior model (spatial outliers β_s). Estimation at resolution k is now expressed as the global minimization of:

$\mathcal{H}^k(w^k + dw^k, \delta^k, \beta^k; f^k) \triangleq \mathcal{H}_1^k(w^k + dw^k, \delta^k; f^k) + \alpha\, \mathcal{H}_2^k(w^k + dw^k, \beta^k)$.

The optimization problem at each resolution level will be performed by a hierarchical "constrained" exploration of the configuration space [5]. The optimization is lead through a sequence of nested configuration subspaces, giving at each resolution a cascade of optimization problems of reduced complexity (multigrid)(see figure 1.b), solved in terms of iteratively reweighted least squares (WLS) within a coarse-to-fine strategy.

4. PARALLELIZATION

The first step of the parallelization is to separate work between a master and its slaves, and to evaluate and reduce communications to a minimum. The master process will perform initializations, synchronization and control, while slaves will do the calculations, each slave dealing with a part of the image (typically a group of contiguous lines). The first initializations, including the building of the multiresolution pyramid are performed by the master. Initialization results are transmitted to slaves. Data and spatial outliers are computed from local variables while the WLS minimization needs values of the neighboring points. Extra communications need to be established between slaves, at each loop iteration, to exchange spatial outliers and incremental velocity with their respective neighbors. Each slave will calculate a local convergence criterion, and send it to the master, computing the global convergence criterion. If convergence is assumed, convergence loops are over and slaves give their results back to the master. Table 1 shows the structure of a slave.

```
.init
.for resol = K to 0
.   for grid = L to 0
.      repeat
.         data and spatial outliers
.         repeat
.            WLS
.         until convergence
.      until convergence
.   end for
.end for
```

Table 1: slave structure

5. PROBLEM AND CAUSES

The parallel efficiency of this first version is very disappointing. Several causes can be found for this lack of efficiency : communications are still huge and very frequent regarding to the computation load; communications in themselves are slow and unstable on an Ethernet network comparatively to networks interconnecting processors of massively parallel machines; each

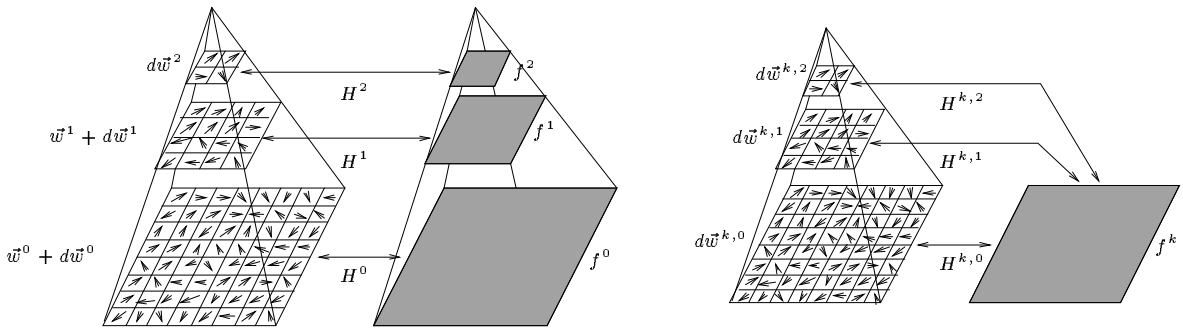

Figure 1: (a) Structure of the multiresolution MRF model ($N = 2$) – (b) Multigrid structure for relaxation at level k ($L = 2$)

exchange between slaves inside convergence loops induces synchronization processes, equalizing the speed of processors to the speed of the slowest; and the work load in the different areas of image is inhomogeneous.

6. SOLUTIONS

Several solutions have been developed to cope with these problems. Emission strategies of PVM have been modified, in order to open direct task-to-task communications and to reduce data type conversions during communications.

At the end of each convergence iteration, each slave must wait for the master computed global convergence condition. The faster the slave is, the longer the waiting is, and the more computing resources are wasted. A load balancing can be performed at the end of each grid loop. The only extra communications will be the emission, to the other tasks, of the motion velocities of the lines belonging to their new computing area after the load balancing. Each task will deal with a number of lines of the image proportional to its own speed.

7. FIRST RESULTS

Results have been computed with a set of Sun Sparc 5 workstations, linked through an Ethernet network, on a set of images called "cars" (cars moving on a parking lot). The first results are presented in figure 2.a under the label "synchronous algorithm".

Plots up to 7 slaves show a decrease in computing time as the number of processors increases. Results from 9 slaves show that overall computing time increases with the number of slaves. This totally unwanted behavior is due to the fact that the communications increase with the number of slaves. From 8 slaves, the computing power gain is overbalanced by the communication cost. Parallel efficiency ($\frac{\text{time on 1 proc.}}{N \text{ (time on } N \text{ proc.)}}$) corrected by Amdahl's law is shown on figure 2.b and expressed the sharp decrease from the 7^{th} slave.

8. ASYNCHRONOUS MOTION ESTIMATION

The modifications exchanged by the slaves are usually quite small, and one or two iterations delay in the exchange will only slightly increase the number of iterations before the convergence of the WLS algorithm but will reduce the communication costs and most of all the synchronization delays. The process becomes *asynchronous*. Exchange rules will be : "send the lines to the neighbors and **if** a message is present **then** Receive it **else** consider that the line is unchanged". Convergence criterion will be moved from global to local. Each slave will estimate if convergence is obtained for its own set of lines, and then send speed results to the master for the load balancing.

This can lead to early end of slaves dealing with less active areas of the image, giving to the load balancing the possibility to balance the processor power disparity and also the work load disparity among the image. The changes during an iteration are usually quite few. Transmitting a whole line if only two or three points have changed may not be relevant. A new communication strategy can use a threshold and only transmit the neighbors if significant changes happened in the line.

The load balancing has been modified to have a good correction capability during the first levels of the grid loop, to counteract processors differences, and to be smoother in the last levels because the only new differences will come from the image itself (areas quiet concerning low frequency motion but active in the high frequencies (for example the motion of the leaves on a tree)). Because these modifications are not easily predictable, changing need not be too sharp (this unpredictability is responsible for the better performance of the synchronous algorithm compared with the asynchronous with 5 slaves).

9. RESULTS

Both methods, synchronous or asynchronous, show similar results when a few slaves are involved. Asynchronous method is faster when the number of slaves increases. At a low number of slaves, the increase of the number of iterations before the convergence balances the synchronization delays gain. When the number of slaves increases, the communication weight in the asynchronous algorithm increases less than in the synchronous one. Asynchronous algorithm presents then a more efficient behavior.

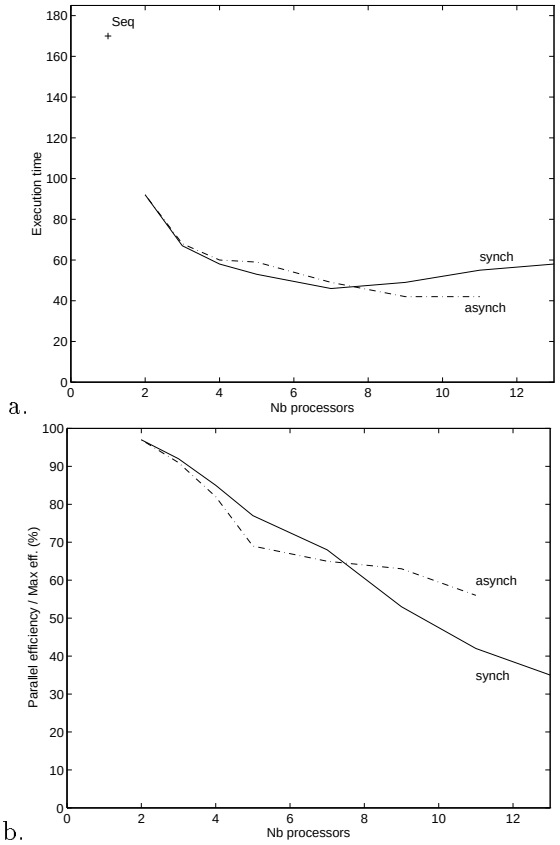

Figure 2: Computing time - Parallel Efficiency

10. CONCLUSION

This paper aimed at the parallelization on a common network of workstations (thus at no additional hardware cost) of an algorithm more suited to massive parallelization. Usual networks such as Ethernet offer a low and irregular bandwidth while processors can have an important computing power. Efforts have been devoted at suiting the algorithm to these characteristics. A first version of the algorithm has been developed, suffering from efficiency limitations when the number of slaves increases. An new asynchronous algorithm has been derived from the original one, giving good efficiency and offering important acceleration factors.

11. REFERENCES

[1] PVM acquisition. pvm@msr.EPM.ORNL.GOV, netlib@ ORNL.GOV.

[2] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions and the bayesian restoration of images. *IEEE Trans. Pattern Anal. MAchine Intell.*, 6(6):721–741, 1984.

[3] W. Hackbusch. *Multigrid methods and applications.* Springer-Verlag, 1985.

[4] F. Heitz and P. Bouthemy. Multimodal estimation of discontinuous optical flow using markov random fields. *IEEE Trans. Pattern Anal. MAchine Intell.*, 15(12):1217–1232, 1993.

[5] F. Heitz, P. Perez, and P. Bouthemy. A multiscale minimization of global energy functions in some visual recovery problems. *CVGIP : Image Understanding*, 59(1):125–134, 1994.

[6] B. Macq, M.P. Queluz, and B. Simon. Very low bit-rate image coding on adaptive multigrids. *Signal Processing : Image communications*, (6), 1995.

[7] E. Mémin, F. Heitz, and F. Charot. Efficient parallel nonlinear multigrid relaxation algorithms for low-level vision applications. *Journal of Paralell and Distributed Computing*, 29(1):96–103, 1995.

[8] E. Mémin and P. Pérez. Multiresolution markov ramdom field and multigrid algorithm for discontinuity preserving estimation of the optical flow. In *SPIE conference on Neural, Morphological, and Stochastic Methods in image and Signal Processing*, 1995.

[9] D.W. Murray, A. Kashko, and H. Buxton. A parallel approach to the picture restoration algorithm of geman and geman on an SIMD machine. *Image and Vision Computing*, 4(3):133–142, 1986.