A self-tuning system for real-time Optical Flow detection

Giovanni Adorni, Alberto Broggi, Gianni Conte, Vincenzo D'Andrea
Dipartimento di Ingegneria dell'Informazione
Università di Parma
Parma, I-43100, ITALY

Abstract **The work presented in this paper is aimed towards the real-time analysis of image sequences acquired from a moving vehicle. Our goal is to extract the optical flow field from the image sequence. The information obtained will be used to detect obstacles in front of a moving vehicle and to compute the time to impact for approaching objects. These requirements mean that special care must be taken with the response time of the algorithm.**

In order to obtain a fast response, a special-purpose massively parallel architecture was used. Furthermore, the algorithm uses an heuristic approach rather than an analytical one.

In the approach presented here, the detection of optical flow is driven by a high level process, by tuning algorithm's parameters.

I. INTRODUCTION

The interpretation of visual motion is the process by which a description of the environment in terms of objects, their three dimensional shape, and their motion through space is constructed on the basis of the images reaching the eyes. But how we perceive the huge amount of information that reaches our eyes? Some researchers consider perception as the mind's window of the world: its purpose is to recognize and report cues, objects and relations between them that are important to the human being. It is this "perceptual connection" between the real world and our idea of the world that ensures that our thoughts are meaningful, and that they carry information about the world. What is crucial in this view is how our perceptual apparatus organizes sensory data. However, sensory data underdetermines the scene structure. The pixels of the image array, by themselves, are not able to determine anything. To obtain any assertion about the viewed scene, knowledge of the image formation (or sequence of images) and the structure of the real world is required.

. This means that perceptual processes are inferential, *top-down*, where the premises for each inference are based on mental construction rather than on what is presented to the eyes and constrained by the biology of the visual system. Other theorists believe perception is largely data driven, *bottom-up*, in every day life. In contrast to the previous two classes of perceptual theories, known as *indirect* and *direct* perception, another class of theory has been proposed: *directed* perception, which focuses on the idea that observers are directed at and select from multiple sources of information that may specify a given stimulus.

The proposed approach, influenced by the previously discussed perceptual models, is based on a bottom-up data-driven process, but can cooperate with a top-down expectation-driven processes. The purpose of the cooperation between top-down and bottom-up processes is to reduce the computational load, and to tune up the optical flow computation by by making use of external world knowledge.

In this paper we give a contribution to the problem of motion perception and discuss an approach to Optical Flow (OF) detection. Due to the constraints imposed by real-time applications based on low-cost systems, a special-purpose massively parallel SIMD architecture (PAPRICA) has been used and the discussed algorithm has been designed to be implemented on such an architecture.

The next section will briefly present PAPRICA computer architecture which will be used as the hardware platform to run the massively parallel part of this algorithm; section III will show some OF algorithms presented in literature, while section IV will focus on our own algorithm. Finally section V will draw some conclusions.

This work was partially supported by CNR Progetto Finalizzato Trasporti under contract number 91.01031.PF93.

II. THE TARGET ARCHITECTURE: PAPRICA

PAPRICA system (PArallel PRocessor for Image Checking and Analysis, [2, 5]), based on a hierarchical morphology computational model has been designed as a specialized coprocessor to be attached to a general purpose host workstation: in the current implementation it is connected to a SUN workstation through a VME bus. PAPRICA is made up of 4 major functional parts: the Program Memory (storing up to 256k of instructions), the Image Memory (up to 3 MBytes), the Processor Array (PA) and the Control Unit.

The first prototype of the PA is composed of an array of 4×4 ICs, each of them containing a sub-array of 4×4 Processing Elements (PE). In the current implementation, the PA is a 16×16 square matrix of 1-bit PEs each one with full 8-neighbors connectivity. Each PE has an internal memory composed of 64 bits; a single 16-bit memory fetch takes 250 ns, while the PA cycle time is 500 ns. The maximum computational speed that can be achieved with the present implementation is 2 Mpixel/s.

PAPRICA image memory can be rearranged run-time, using special instructions which set the image height, width and depth. Moreover, it is possible to take advantage of the fact that for each parallel computation $2Q^2$ sequential accesses to the image memory are required (where Q is the linear dimension of the PA, i.e. 16 in the current implementation), fetching and storing back the data in a useful way. In fact, the data to be transferred into the PA can be not logically adjacent to each other in the image memory, allowing to undersampling of the image or an increase in its resolution. This behavior is controlled by a set of registers which can be altered run-time. The possibility of being able to reduce and increase image dimensions allows a very efficient use of PAPRICA as multiresolution architecture. Moreover, simple software algorithms also allow the simulation of any kind of pyramidal interconnections between the different layers.

III. APPROACHES TO OPTICAL FLOW DETECTION

The goal of this work is to approach real-time performances on PAPRICA architecture. In this section we will briefly analyze some algorithms that seem to be the most promising with respect to our goal.

The algorithm proposed by Tretiak and Pastor [11] is based on the assumption that when the scene in two subsequent frames does not change, the brightness gradient is constant in space and time. Thus, it imposes the time derivative of the gradient as zero, and it finds the points in the image where this hypothesis is not true. In these points, it derives information on the spatial modifications, and it gets the OF vector components. To do that, second order derivatives are needed. Although it is possible to simplify the computation of these derivatives in some special cases (see, for example, [10]), at least a 5 pixel neighborhood is needed to collect sufficient information. Moreover, the Tretiak and Pastor algorithm makes a large use of temporal filtering: namely, to obtain the result for a particular frame, the simultaneous analysis of a number of previous and successive frames is needed. In this way, the time required for the whole computation of the OF in a particular frame is only given by the computational time, plus the intrinsic delay due to the need for several successive frames.

This approach follows the assumption that the function describing image brightness distribution is continuous. Although mathematically correct, the main problem of this approach is the need for sequences of frames with very smoothly varying scenes, (i.e. with uniform and simple movements). This requirement is equivalent to having a very high sampling frequency: two subsequent frames need to be very close to each other, and this imposes very fast processing. We tested this algorithm on a CM-2 Connection Machine [6], with noisy images grabbed from a moving vehicle in outdoor environments. The results obtained were not encouraging, taking into account the spatial complexity of the algorithm, given by the memory requirements to store five 8-bit frames and some temporary results.

Since one of the characteristics of the architecture used in our experiments (PAPRICA) is the small amount of memory for each processing element and the possibility to operate only on single-bit data, with extremely simple operations, this approach is too complex and leads to slow computational speed. It has been implemented on PAPRICA, by trying to use smaller images: from 256×256 to 64×64, and from 8 bits/pixel to 4 bits/pixel. The experimental results confirm that the approximations, which cause the processing image to be more quantized, are conflict with the requirements of the algorithm, i.e. smooth images [9].

Another approach has been tested [7], an iterative one, which requires only two images and thus a restricted amount of memory. Moreover it is based only on algebraic computations, which are much more suitable for PAPRICA than the previous ones. The main problem of this implementation is the iterative approach, which is typically very time-consuming. Furthermore, the authors use images with an artificially added noise of 1% and very low relative speeds, which is very far from the characteristics of the natural images grabbed from a moving vehicle. It has been implemented and tested on a CM-2 Connection Machine with images taken from a camera installed on a moving car, and also in this case, the results were not encouraging.

Finally, the approach proposed in [3] was considered since it has the capacity of being invariant to brightness changes during the sequence. This property is very useful when analyzing image sequences in which the change in global brightness is very frequent (as, for example, road traffic scenes). It is based on a double analysis of the sequence: the first one is used to get the brightness variations in the complete sequence; the second to get the OF components, knowing the brightness variations and trying to compensate them. Unfortunately this approach is very time-consuming and leads to high computational complexity. Moreover the results shown by the authors (in [3]) refer only to synthetic and laboratory generated image sequences.

IV. A SIMPLE OPTICAL FLOW DETECTION ALGORITHM

All the previously analyzed algorithms and most of the algorithms in literature (see, for example, [8] for a general survey), are based on the assumption that the frames can be described by analytical functions, and thus derivable. Since this is just a useful approximation (each frame is in fact a digitized image, and thus intrinsically discontinuous), the general strategy is to preprocess each frame with smoothing filters in order to make them derivable. Unfortunately, it is the presence of a physiological discontinuity of the brightness distribution in a pixel that provides (within limits) the solution to the problem of correspondences. Due to the architecture of PAPRICA , the problem of correspondences should be approached in terms of a small number of local operations but with the guarantee of ambiguity detection in the results.

To do this, a simple pixel-to-pixel correlation has been implemented. It is based on a neighborhood size varying Cellular Automaton [4, 1], which is performed by scanning the neighborhood in a spiral. This special scanning of the neighboring pixels allows the evaluation of the possibility that matching is caused by noise or by the correct determination of the corresponding pixel.

First of all, a weighted average between the pixel-to-pixel correlations was implemented, using a gaussian function proportional to the inverse of the distance from the central pixel as weight . Unfortunately this kind of approach needs floating-point computations, which are not suitable for PAPRICA's architecture. This reason led to the replacement of this analytical approach, with a heuristic one: the determination of the OF components is done directly by searching in the neighborhood for a pixel with a similar brightness value. In literature, the possible influence of noise is normally reduced by extending the correlation to a larger window. In contrast, in this implementation the discarding of noisy results is accomplished by post-filtering, which has the purpose of eliminating the OF vectors that have a different direction to the local average. This choice simplifies the correlation step and introduces a low-cost post-processing average operation. The heuristic approach is based on the following statement: if during scanning only one maximum of the correlation is found in the whole neighborhood, the OF can be determined. On the other hand, when more than a single maximum is found, the OF determination is only possible if the distance between the central pixel and the first two closest maxima are sufficiently different. The OF is computed using the closest maximum to the central pixel. Otherwise, the OF vector associated to the central pixel is considered ambiguous and discarded.

Another key parameter of the algorithm is the number of processing image bits/pixel. In fact, a pre-processing operation can be performed to reduce the initial 256 gray levels of the input image. The higher it is, the more sensitive to noise the result becomes, and the lower the number of matching pixels. Obviously this is a rough technique to increase the signal-to-noise ratio, but it has been proven to be the one that achieves the best performance on PAPRICA architecture, in terms of computation complexity (reduced to a virtual shift of data), and results.

The spatial complexity of the presented algorithm in terms of memory requirements is very low: for a single OF field computation only 35 bits per processing element must be allocated, both for frame storage and for temporary storage. As a comparison, the application shown in [10] needs at least 1000 bits per processing element.

A. Algorithm Parametrization

In order to have an algorithm that produces significant results, two subsequent frames must be similar enough to let the algorithm look for the correspondent of each pixel in a small neighborhood. Recalling that our camera is installed on a moving car, this characteristic is obtained only if the two frames are sampled in correspondence to sampling-points on the road that are sufficiently close to each other. Two subsequent frame samplings must be executed within a space interval denoted with s' (see figure 1).

For a given set of parameters p', the algorithm can provide the output frames at a speed of F frames per second:

$$F = f(p') = [frames/s] \tag{1}$$

In order to have a set of parameters that enables the algorithm to produce meaningful results, the spatial sam-

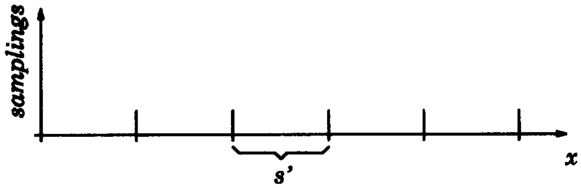

Figure 1: Spatial samplings

pling distance of two successive frames must be smaller than s'. If the vehicle moves at a speed v, the spatial distance s' is covered in a time interval t', obtained from

$$t' = \frac{s'}{v} \tag{2}$$

Knowing that F frames per second can be processed, in order to produce meaningful results at least one frame must be sampled and processed in the time interval t'. Thus,

$$F \cdot t' \geq 1 \tag{3}$$

Using equations (2) and (3), it is possible to obtain a threshold equation:

$$F \geq \frac{v}{s'} \tag{4}$$

Equation (4) subdivides the (v,F) plane into two regions: the upper one is the region in which the algorithm works correctly, while the lower one shows the region in which the algorithm does not produce meaningful results.

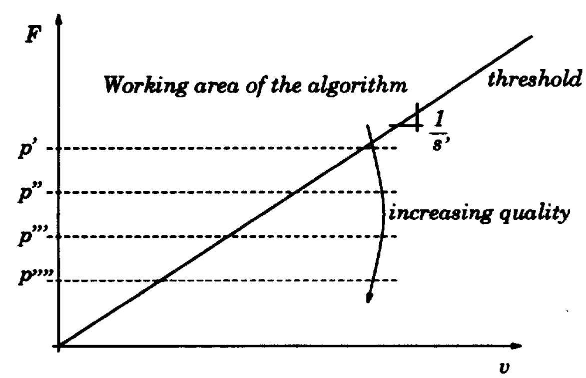

Figure 2: Algorithm performances: v vs F

Since F is independent of v, the iso-performance curves (that is the curves obtained with a specific set of parameters p) are represented in figure 2 as straight lines (with slope 0).

Defining S

$$S = \frac{1}{F} \tag{5}$$

as the number of seconds required for processing a single frame, the threshold function becomes

$$S \cdot v \leq s' \tag{6}$$

which is an hyperbole, as shown in figure 3.

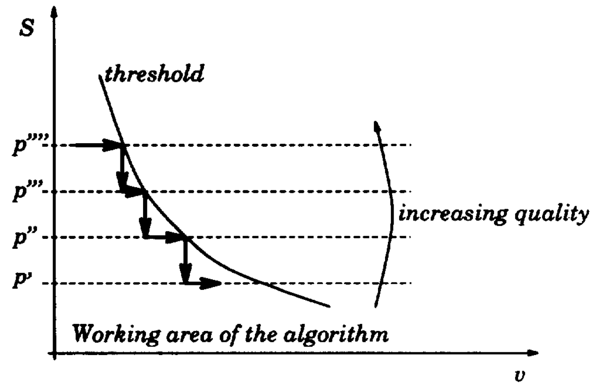

Figure 3: Algorithm performances: v vs S

The main characteristic of this parametric approach is that it allows its parameters to be modified according to external requirements. Figure 3 shows that, in order to decrease the time required to process a frame, the parameters must be changed toward a more approximated solution. The arrows in Figure 3 show the working points of the algorithm during vehicle acceleration.

Using equation 6, it is possible to estimate the maximum vehicle speed as a function of the algorithm performance. The set of parameters includes a 7×7 Cellular Automaton neighborhood, a subsampling value of 1:4 (resulting in a 256×256 image) working on a 8 bits/pixel image. With this restrictive choice of parameters, the algorithm can compute the optical flow in $\simeq$350 ms. Using spatial sampling of 1 m, the vehicle should move at a speed less than 10 km/h.

B. Self-tuning of the Algorithm

The architecture of the proposed approach is sketched in figure 4. The block computing the Optical Flow takes as input an image (taken by the camera) and the set of its operating parameters.

The time needed for a whole computation of a single OF vector field is a function of all the algorithm parameters and its performance can be tuned according to specific requirements of speed and/or output quality (precision); the principal key parameters that control its behavior are:

- the dimensions of the input image: an initial subsampling can reduce the amount of data to be processed;

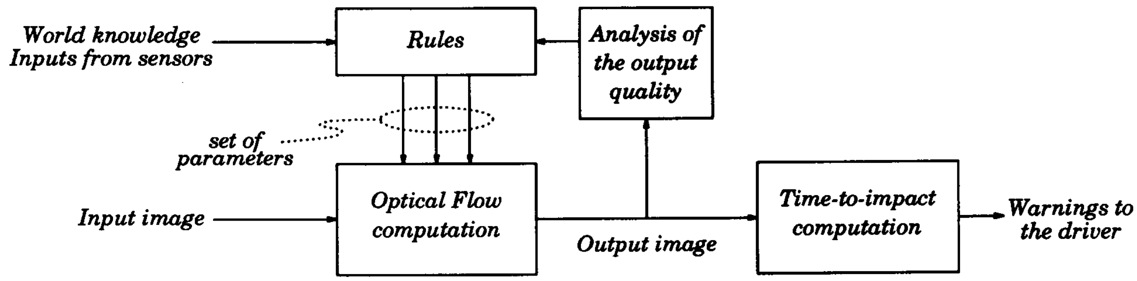

Figure 4: Architecture of the proposed approach

- the number of bits/pixel: since PAPRICA is made of single-bit processing element, there is no need for an image to be made of a particular number of bits/pixel, such as 4 or 8;
- the dimension of the neighborhood needed to solve the problem of correspondences;
- a set of thresholds to pre- and post-process each frame.

Figure 5 shows the results of an implementation of the discussed algorithm with different sets of parameters, using the PAPRICA simulator running on a CM-2 Connection Machine.

The tuning of the algorithm is performed by means of a rule-based system (top-down process), which, is able to change: 1) image resolution, 2) sampling rate, 3) number of gray levels, 4) Cellular Automaton neighborhood dimensions, 5) pre- and post-processing set of thresholds, on the basis of heuristic knowledge and external information (e.g. the speed of the vehicle, the illumination of the scenario, etc. etc.). This means changing the curve family in order to maintain real-time performance while reducing the output quality of the result.

An example of a heuristic rule is:

```
IF
   "the speed of the car increase" AND
   "the iso-performance curve is P(i)" AND
   "the number of frames/s is under the
     threshold"
THEN
   "move to the iso-performance curve p(i-1)".
```

The system structure shown in figure 4 highlights the integration between the top-down and bottom-up approaches. The tuning of the OF computation is performed by a rule-based system according to a measurement of the quality of the computation. The measurement will be mostly based on statistical considerations (e.g. the average direction of the OF vectors, the number of significant vectors per area unit, the average length of the vectors,...).

The rule-based system also makes use of *world knowledge*. This knowledge contains information on the road being traversed, taken from a geographical database. The system will know the approximate position of the car, and the parameters can then be adjusted, for example, according to the urban or extra-urban environment.

We are currently experimenting the OF algorithm in several different conditions, in order to set up rules relating the algorithm parameters to the measurements of external sensors (e.g. the speedometer).

V. CONCLUSIONS

In this paper a fast self-tuning algorithm for OF computation has been briefly discussed. The algorithm has been designed to be implemented on the massively parallel SIMD architecture, PAPRICA, as a part of a complex vision system for the processing of road traffic images. This research is part of a Eureka project, PROMETHEUS, which aims to improve traffic safety, by increasing the quality of information provided to the car driver. To this end, a computer vision system plays an important role, since most of the information available when driving is of a visual nature. Such a vision system, integrated into a "smart" sensor, should be able to provide the driver with many different kinds of information.

From experimental results obtained using a PAPRICA simulator (running on a CM-2 Connection Machine) the described approach seems to be very promising. At the moment of the writing this paper the first PAPRICA boards are in operation and we expect to test the algorithm in the very near future.

Because of its processing element virtualization mechanism, PAPRICA can efficiently simulate multiresolution architecture. One of the main future improvements to the system will be the extension of the algorithm towards a pyramidal approach, computing OF vector field

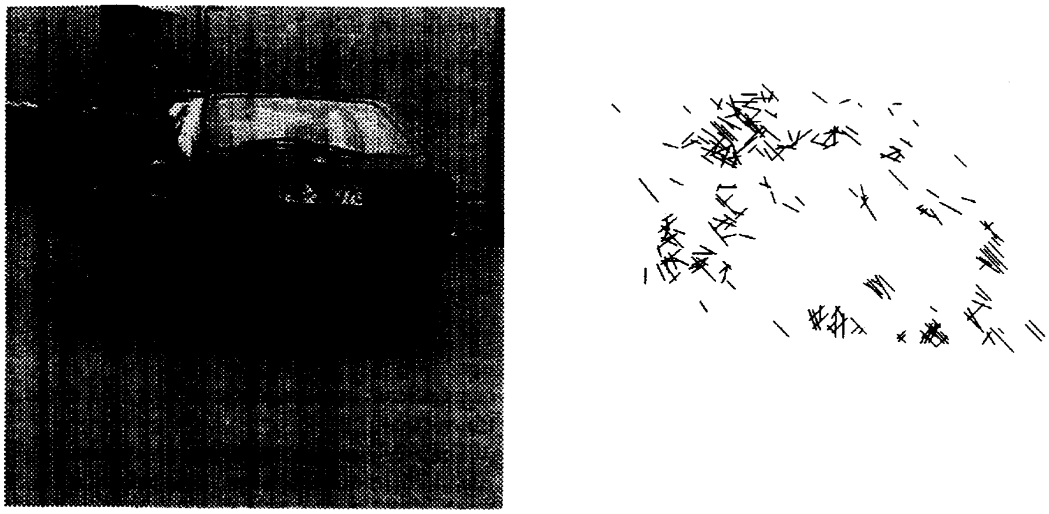

Figure 5: Optical flow computed by the proposed algorithm

at different resolutions. Higher resolution computations will then be used to confirm or reject lower ones. The image is subsampled and a coarse OF vector field extracted, which is then used to drive simpler OF detection using images with increasing resolutions.

ACKNOWLEDGMENTS

The authors would like to thank G. Maris and A. Folli for their support during the implementation of the algorithm.

REFERENCES

[1] A. Broggi, V. D'Andrea, and G. Destri. Cellular Automata as a Computational Model for Low-Level Vision. *International Journal of Modern Physics C*, 4(1):5–16, 1993.

[2] A. Broggi, V. D'Andrea, and F. Gregoretti. A low-cost parallel VLSI architecture for low-level vision. In *MVA'92 - IAPR Workshop on Machine Vision and Applications*, Tokyo, Japan, 1992. International Association for Pattern Recognition.

[3] J. Ducan and T. Chou. On the detection of motion and the computation of optical flow. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 14(3), 1992.

[4] D. Farmer, T. Toffoli, and S. Wolfram, editors. *Cellular Automata*, Amsterdam, March 1983. North-Holland.

[5] F. Gregoretti, L. M. Reyneri, C. Sansoè, A. Broggi, and G. Conte. PAPRICA - A Dedicated Processor for Morphological Image Analysis. In *Proceedings 7th International Conference on Image Analysis and Processing*, Bari, Italy, September 20-22 1993.

[6] W. D. Hillis. *The Connection Machine.* MIT Press, Cambridge, Ma., 1985.

[7] B. Horn and B. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17(1-3):185–204, 1991.

[8] D. Murray and B. Buxton. *Experiments in the Machine Interpretation of Visual Motion.* MIT Press, 1990.

[9] M. Snyder. On the Mathematical Foundations of Smoothness Constraints for the Determination of Optical Flow and for Surface Reconstruction. In *Proceedings DARPA Image Understanding Workshop*, San Mateo, California, May 1989. Morgan Kaufmann Pub. Inc.

[10] M. Tistarelli. Computing Optical Flow: a Real Time Application of the Connection Machine System. Technical Report V89-1, Thinking Machines Corporation, June 1989.

[11] O. Tretiak and L. Pastor. Velocity estimation from image sequences with second order differential operators. In *Proceedings 7th IEEE International Conference on Pattern Recognition*, pages 16–19, 1989.