Accepted to IPA99 Manchester july 1999

Segmentation of moving objects at frame rate: a dedicated hardware solution

R. Cucchiara[+], P. Onfiani[+], A. Prati[+], N. Scarabottolo[*]

[+]University of Modena, Italy [*]University of Crema, Italy

Many works in image processing concern segmentation of moving objects in sequence of images. This problem is particularly critical, since it represents the first step of many complex processes of computer vision, for applications like object tracking, video-surveillance, monitoring, and autonomous navigation. In such applications, both real-time and low-cost requirements should be satisfied.
To this aim we propose a dedicated hardware solution, based on reconfigurable logic, that provides motion detection and moving objects segmentation at frame-rate.

INTRODUCTION

Techniques for motion extraction from images come from two opposite approaches. From one side, many analytical methods of motion analysis have been formalised: examples are the studies on optical flow (Beauchemin and Barron (1)), or the extraction of moving features (Kanade and Tomasi (2), Smith and Brady (3)); these methods are able of extracting not only moving points but information of the motion direction and velocity too, and are generally very expensive in terms of computational time. For example in Liu et al. (4), a real-time motion detection system is proposed, but running on an HyperSparc workstation with it needs a Datacube MV200. At the other side, many proposed approaches have been developed only in function of specific real-time applications, and therefore prefer empirical and approximated solutions. Examples are methods based on the analysis of single columns of pixels for measuring vehicular flow on roads, called Inductive Loop Emulators (Fathy and Siyal (5)), or systems that make use of road markings in order to easily evaluate moving objects with respect to the fixed background (Bouzar et al. (6)). These video-based systems works at pixel-level only and are not able of providing detailed information on individual vehicles, thus lacking generality and flexibility.
Our proposal can be viewed as a trade-off between the goal of achieving object segmentation with very simple processes (easily portable in hardware) and at the same time the aim of processing the whole image in order to extract the complete information of the moving objects and their shape. The goal is to provide a hardware solution of the problem of *detecting moving objects* in outdoor scene at real-time.
Therefore the aim of this work is the development of a system for video-surveillance and object tracking with these issues: *simple processes*, in order to be able to develop a cheap and reliable VLSI implementation; *real-time issues*, to achieve a frame-rate processing, necessary for video-surveillance applications; *flexible* approach and possibly reconfigurable in dependence of the final application; *on-site implementation* without the necessity of an installation of cumbersome general-purpose PCs where vehicles have to be detected.

To achieve all these issues a prototype based on SRAM-based FPGAs has been developed.
The approach we follow, oriented to surveillance and traffic control applications, extracts moving points from images with simple image processing techniques, segments them in order to obtain moving objects; the location and identification of object will be used in a further tracking step (Koller et al. (7), Barattin, Cucchiara and Piccardi (8)). In particular we adopt a spatio-temporal filtering that consists in the integration of motion information from sequences of frames with the information of gray level variation in each single frame.
In the paper we outline the proposed approach for extracting moving objects, that in the specific road context can be classified as vehicles and we describe the hardware solution based on reconfigurable-hardware. Finally we present performance results of the developed prototype.

THE PROPOSED APPROACH

Moving objects in an outdoor scene can be perceived by an observer since both motion and luminance contrast concur to pop the object shape out the background.

Accordingly, many approaches exploit differential computation with both spatio and temporal filtering. A possible promising solution (8), exploits the following steps for moving object extraction:

1) detection of moving points by p difference on three consecutive frames
2) detection of high contrast points i points with high gradient, as possi moving points;
3) perform a Moving Edge Closing morphological closure between movi strong edges in order to extract movin

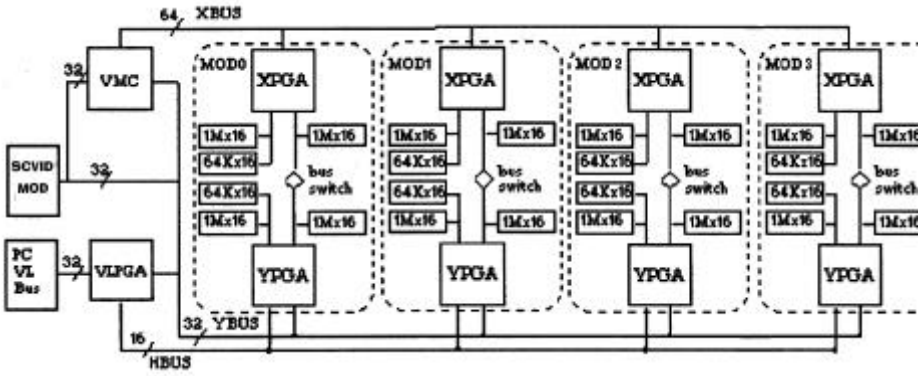
The algorithm for extracting moving poin the difference of three consecutive Yoshinari and Michihito (9)): we adopt since we proved that is particularly robust to noise due to camera movements and avoids the detection of very small moving objects in the scene (such as tree leafs, reflections..).

The three steps in (8) are followed by other processes for obtaining separate objects even in presence of occlusion: the objects are finally classified as moving vehicles according to some rules and a rule-based tracking system is proposed.

Independently from the adopted high-level symbolic system, the low-level system must perform moving objects segmentation at very high speed, in order to meet real-time requirements of applications. Therefore, most or all frames must be processed.

From these requirements, the need of a system able to segment moving object at frame rate arises. The previous considered three steps take about one second tested on a high performance PC for images just stored in central memory and thus without considering transfer time for the frame grabber acquisition. This time, even if it is not too critical, is still far from real-time requirements. Moreover in many real applications, the adoption of a standard PC is not affordable for many reasons, first of all the cost requirements: in many distributed applications such as road traffic control a suitable solution should equip all traffic-lights of an intelligent camera able of detecting and measuring the vehicular movement.

Finally an alternative could be the real-time transfer of all frames from the road to a possible processing center: but also this way is not affordable for the current costs of transfer bandwidth (for example, in the traffic control system mounted and running in Bologna, Italy, cameras transfer rates are of 0.2 frame/sec only).



HARDWARE IMPLEMENTATION

In this context, the main contribution to this work is to propose a robust approach and its hardware implementation provided with Field Programmable Gate Arrays, that answers the requirements of high integration, and possible low-cost.

The developed prototype is based on the Gigaops G800 board (Giga Ops Documentation (10)).

In this working environment, we have implemented different versions of differential algorithms for the moving points extraction with two-frame difference, three-frame-difference and difference with background approaches (9) in order to compare results in various external conditions. As well as moving point extraction, we perform concurrently edge detection and Moving edge closure at real-time.

The prototyping board we used is the GigaOps *G800* Spectrum board (10), schematically shown in figure 1.

In figure 1 it is possible to notice the main blocks of this system. The actual computation is performed by pairs of Xilinx XC4010E FPGA's, connected in modules called XMOD's: in Fig. 1, four modules (MOD0 thrug MOD3) are shown. Both these FPGAs have two memory ports: they communicate through a bus switch on the first memory port. Moreover, there is a module called SCVIDMOD, that decodes/encodes video signals (PAL or NTSC). Furthermore, an input FPGA (here called VLPGA) is connected to the local bus of the PC hosting the board. An output FPGA (here called VMC) is connected to SCVIDMOD. Three main busses allow connections among the various blocks of the board.

We developed the system in VHDL language and compiled it with Synopsys Tools (Synopsis (11)) . The netlist file obtained has been translated in bitstream in order to be downloaded into FPGAs at execution time. Once bitstream has been produced it can be downloaded to the hardware every time we need to use it.

Figure 1: Block diagram of the GigaOps G800 board

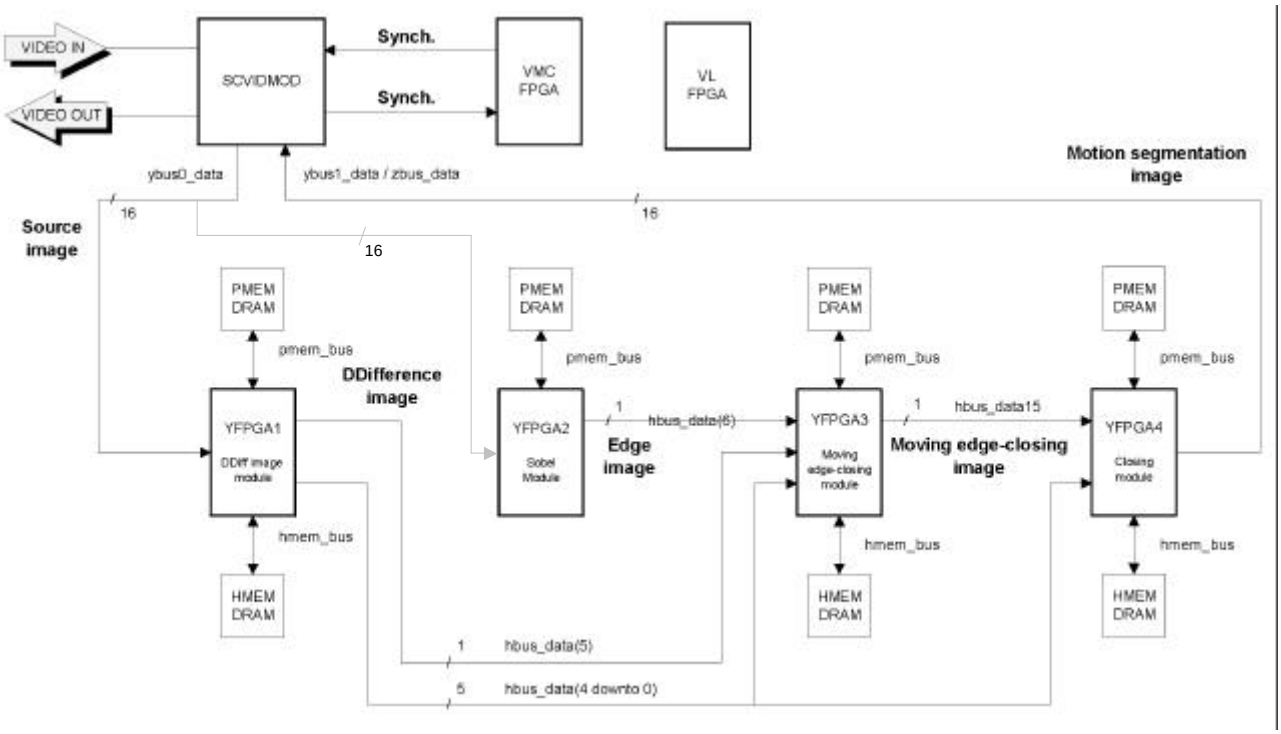


Figure 2. Data path for final prototype

VEHICLE DETECTION

All operations are executed in pipeline with delay lines for performing mask near-neighbour algorithms (e.g. edge detection) exploiting the synchronism clock of the acquisition system.
In our approach, target extraction is based on *spatio-temporal* segmentation: "temporal", because it exploits information on moving points; "spatial", because it performes convolution to exploit luminance variations in a 3x3 near-neighbour mask to select edge points. We define a suitable *moving edge closure*, in order to obtain a close contour of a moving object. This algorithm correlates moving points (that is detected by the double-difference algorithm) with high gradient point (extracted with a standard Sobel operator).
Temporal and spatial filtering have to be performed simultaneously. Therefore we exploit the data parallelism available in the G800 board in order to meet the real-time constraints required by the application.

In figure 2, data path is shown. The image coming from a standard camera and grabbed from decoder (performed by the SCVIDMOD) flow both to YFPGA1 and to YFPGA2. Through the former we obtain the double-difference image, while the latter performs edge detection. Results are sent to the YFPGA3 through hbus_data(5) and hbus_data(6), respectively. The former contains the binarized information of the moving points (using a hysteresis thresholding), the latter contains the edge image, binarized too.
The lines hbus_data(0)-hbus_data(4) reach each module to synchronize them through a semi-frame counter.
YFPGA3 performs the *moving-edge closure* above described, exploiting information from moving and contours points. The final results of this operator are passed to YFPGA4 through hbus_data(15) to be able to perform a further morphological closure with four closing steps.
Finally, in the current prototype, results are sent to encoder (i.e. SCVIDMOD) in order to be displayed on the monitor.

The final morphological closure is an optional step that can be useful for providing closed contours of moving objects. However this iterative operation is time consuming: therefore total throughput of the system has

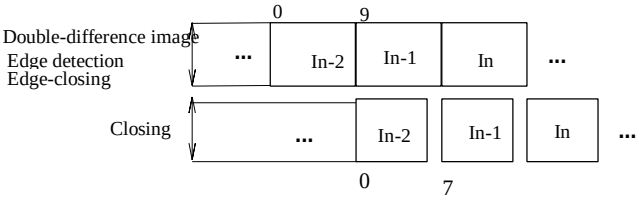


Figure 3. Two-step pipeline

been improved with a two-*step* pipeline, as shown in figure 3.

PERFORMANCE EVALUATION

The current solution of intersection's management in the most of cities equipped with intelligent traffic light controller is based on the usage of inductive loops.

These devices produce only information on the number of vehicles passing over them. However, lack of information (then inflexibility) is not the only drawback of inductive loops. Due to bandwidth of common infrastructure networks mounted in urban environments, data output rate is normally slow (in Utopia system (12) for instance data are updated every 5 seconds) since acquired information on road must reach the traffic light control system.
To increase data output rate, enriching information of the processing system, dedicated hardware solutions are the most promising choices. *ISPDs* (In-System Programmable Devices), such as FPGAs, rely on high integration and reprogrammability to being very useful for rapid prototyping. Furthermore, on-site installation of such devices implies the limitation of the bandwidth in order to meet the real-time constraints, relying on the possibility of local processing in order to transmit synthetic result of processes only, instead of the whole frames.
The european PAL video standard adopts a frame rate of 25 frames/sec., that is one frame every 40 msec. Since PAL standard is interleaved, the above times refer to semi-frames and since double-difference operator needs three whole frames to be performed, in theory we need 240 msec to obtain double-difference image.
But, due to the data-parallelism introduced and to the two-step pipeline shown in figure 3, we are able to overlap operations obtaining YFPGA3's output in 240 msec (edge detection operation is performed in parallel on the source image).
We shall be able to use only three consecutive frames, obtaining frame rate behaviour. Nevertheless, in order to catch movement of vehicles driving from 40 to 60 km/h, we output one frame every five.

As shown in figure 3, the morphological closure must wait the end of moving-edge closure to be able to be performed. Moreover, each step of the closure needs the result of the previous one. With a four-steps closure this means a latency time of 160 msec, to be added to the 240 msec of previous stage of the processing. But since we can pipeline the two steps (onto the last performing step of the double-difference), we can obtain final result in 200+160=360 msec.
Since 200 msec are due to acquisition time, we are able to produce a refreshing time of output image of 160 msec, that is 5 frames/sec. This is enough to obtain a sufficiently good continuity of the movement in the result image. Nevertheless, the 240 msec for double-difference computation are necessary to catch only strong movement of the objects in the scene and to increase robustness of the system.

EXPERIMENTAL RESULTS

Figure 4 show one example frame of four possible video output of our system that, using a standard PAL camera, is able to furnish different video output forms at frame-rate: the sequence of the standard colour image as in Fig. 4a (without any image processing), the moving points as in Fig. 4b, the edge points in Fig. 4c and the moving objects (Fig. 4d), obtained with one-step morphological closure. Therefore a real-time processing is performed and also a severe compression of the images (from colour pixel to 1-bit/point images) keeping only the important information about motion and, at the same time, requiring a very limited bandwidth.
The system has been tested on real road traffic scenes in the cities of Modena and Bologna (Italy). This work is a part of a project sponsored by the Bologna Provincia government for a city control center with vision based traffic monitoring**.**

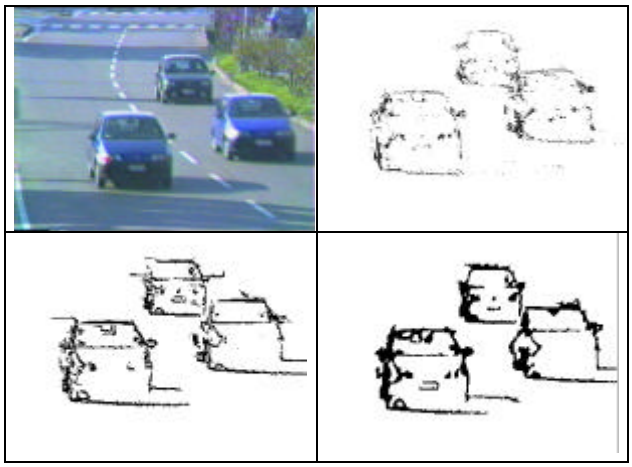


Figure 4. Sample images

CONCLUSION AND FUTURE WORKS

In this paper, we have presented a traffic-control system implemented by using FPGAs. Nevertheless, this system is low-level implementation of a complete urban traffic controller able to track vehicles, count/classify vehicles (for special applications, such as reserved lane for busses, which needs a vehicles classification in loose sense) and to extract extra-information such as turning rates, position and length of the queue, etc.
In (8), the algorithm setup for day time condition and the high level tracking module has been presented. This paper reports the hardware implementation of the low-level algorithm. At the same time, research activities for other day conditions, and in particular at night, are performed (Cucchiara and Piccardi (13)).
Performed experiments show that the vehicle detection under different light condition requires very different image processing algorithms depending on the different visual cue that have to be detected (e.g. vehicle template at daytime and headlight at night). This analysis suggests the exploitation of a reconfigurable low level system able for adaptively change its computational function. In the next future we intend to implementing this dynamically configurable behaviour in hardware,

by exploiting the in field reprogrammability of the FPGAs.

Testing the system upon more sequences (and from different condition, such as rainy, foggy and cloudy) is another goal for the next future.

BIBLIOGRAPHY

1. Beauchemin, S. S., Barron, J. L. . "The computation of optical flow". ACM Computing Surveys, 27, No. 3 (Sept. 1995), 433-466

2. Tomasi, C., Kanade, T. . "Detection and tracking of point features". Technical Report CMU-CS-91-132, Carnegie Mellon University, Apr. 1991

3. Smith S.M. Brady J.M. "Asset-2: real-time motion segmentation and shape tracking" IEEE Trans of PAMI 17(8) 814-820 1995

4. Liu, H., Hong, T., Herman, M., Chellappa, R. . "Motion-Model-Based Boundary Extraction and a Real-Time Implementation". Computer Vision and Image Understanding, 70, No. 1, April 1998, 87-100

5. Fathy, M., Siyal, M.Y. "Real-time measurement of traffic queue parameters by using image processing techniques". IEE Proc. - Image Processing and its Applications n 410 , 450-453 (1995).

6. Bouzar S., Lenoir F., Blosseville J. Glachet R. "Traffic measurement: image processing using road marking". Proc. Of IEE Road Traffic Monitoring and Control n.422, 105-109 (1996)

7. Koller, D. Weber, J., Huang, T., Malik, J., Ogasawara, G., Rao, B., Russel, S. "Towards Robust Automatic Traffic Scene Analysis in Real-Time." Proc. Int'l Conf. Pattern Recognition, 126-131 (1994).

8. Barattin M., Cucchiara R. , Piccardi M. "A Rule-based Vehicular Traffic Tracking System" Proc. Of CVPRIP98 North Caroline 1998

9. Yoshinari K., Michihito M. "A human motion estimation method using 3-successive video frames" Proc. Of Intern. Conf. On Virtual systems and multimedia Gifu, 135-140 1996

10. Giga Operations Corporation. "SPECTRUM™ Reconfigurable Computing Platform – Documentation, Release 3.01"

11 Synopsis, FPGA Compiler User Guide v 3.5 Mountain view (USA) 1996

12 Mizar Automazione, "UTOPIA Urban Traffic Control, Technical Reference Manual", january 1997

13 Cucchiara R., Piccardi M. "Vehicle Detection under Day and Night Illumination" Proc. of ISCS-IIA99 Special session on vehicle traffic and surveillance, June June 1999