SYSTOLIC ARRAY ARCHITECTURE FOR EARLY VISION PROCESSING

J.D. Legat, J.P. Cornil, D. Macq, M. Verleysen

Laboratoire de Microélectronique
Université Catholique de Louvain
Place du Levant 3
B-1348 Louvain-la-Neuve
Belgium

ABSTRACT

This paper describes the architecture and the implementation of a systolic processor dedicated to motion estimation which is one of the most important task in early vision processing. The input image is scanned by a standard video camera with 256 grey levels. Motion computing is based on the optical flow determination. An algorithm to detect multi-motion is proposed and is implemented in real-time in a systolic array.

1. INTRODUCTION

Motion estimation of objects or people from image sequences plays a key role in early vision processing [1]. It is used in dynamic image segmentation and recognition. Relative motion allows mobile robot to navigate quickly and efficiently through the environment [2].

There exist two basic algorithms for determining motion [3]. The first is based on the estimation of special features in the image which are then matched from image to image. This method supposes that the image is first analyzed carefully before computing motion. Psychophysical tests suggest that it is not the case in the human system . The second algorithm uses local gradients of the image brightness to compute the optical flow. In general, the optical flow and the true 2-D velocity field differ. Nevertheless, if strong enough gradients exist, the optical flow will be a good approximation of the velocity field.

Through many types of optical flow estimation techniques have been developed [3], [4], most of them are restricted to global motion estimation [5]. We present here an algorithm for multi-motion evaluation based on optical flow computing. A systolic array which implements in real-time this algorithm for an image coming from a standard video camera, has been designed. The architecture and details about its implementation will be described.

2. OPTICAL FLOW ESTIMATION

Following Horn and Schunk [6], we denote the image brightness at the point (x,y) in the image plane at time t by I(x,y,t). When a pattern moves, the brightness of a particular point in the pattern is constant, so that :

$$\frac{dI}{dt} = 0$$

Using the chain rule of differentiation, we see that :

$$\frac{\partial I}{\partial x}\frac{dx}{dt}+\frac{\partial I}{\partial y}\frac{dy}{dt}+\frac{\partial I}{\partial t}=0$$

If we define the velocity V as $(v_x, v_y) = (dx/dt, dy/dt)$, we have a single linear equation in two unknowns, v_x and v_y :

$$\frac{\partial I}{\partial x}v_x+\frac{\partial I}{\partial y}v_y+\frac{\partial I}{\partial t}=0$$

This equation by itself is not sufficient to determine the velocity flow. It only defines a constraint line in velocity space that has the same orientation as does the edge of the moving pattern in physical space. This ambiguity is known as the "aperture problem" and is illustrated in figure 1. If we suppose that there is only one moving pattern, each edge of the moving object generates a constraint line. The intersection of these lines determines the actual velocity .

3. MULTI-MOTION COMPUTING

If many objects can move independently, it will not be possible to determine the velocity without introducing additional constraints. We can suppose that neighbouring points of a moving object have similar velocities and the velocity field in the image varies smoothly. Based on this smoothness assumption, we introduce the three following constraints :

1. If a moving object at time t has the velocity v(t), it will have at time t+1 a velocity v(t+1) so that :
$$v(t) - \Delta v < v(t+1) < v(t) + \Delta v$$

2. If this moving object at time t is included in a rectangular boundary having the coordinates ((x0(t), y0(t)), (x1(t),y1(t))), it will be included at time t+1 into a rectangle having the coordinates ((x0(t+1), y0(t+1)), (x1(t+1),y1(t+1))) so that :
$$x_i(t+1) = x_i(t)\ (v_x(t) +/- \Delta v) \quad \text{and} \quad y_i(t+1) = y_i(t)\ (v_y(t) +/- \Delta v)$$

3. Only one new motion can appear at the same time

By using these hypotheses, it is now possible to associate each constraint line with a particular motion and to efficiently initiate new motions. This method is illustrated in figure 2 where two objects are moving in different directions. The algorithm has been successfully simulated on synthetic and some natural time-varying images.

4. SYSTOLIC ARRAY ARCHITECTURE

Figure 3 gives the architecture of an 1-D systolic array which implements in real-time the multi-motion computing algorithm. Each node of the array is a full processor handling a particular motion. The first processor receives the image from the camera and some contextual information. Due to the serial output of the video camera, the motion estimation is done pixel by pixel. If the input pixel can be associated with the current motion, a disable signal is transmitted to the second processor. After the pixel processing, the pixel is transferred to the next processor and so on. Each processor works synchronously and has the same architecture. A standard microprocessor controls at a high level all the processing.

5. IMPLEMENTATION

The elementary processor of the systolic array is implemented in a custom integrated circuit. As illustrated in figure 4, the chip consists of 9 processing units : the input pixel unit (IPU) which receives the pixel brightness and some other values from the previous processor (or from the camera for the first processor), the parameter computing unit (PCU) which determines common parameters used by the other blocks, VXU and VYU which compute the velocity in the X and Y directions, BXU and BYU

which estimate the boundary of the moving object in the X and Y directions, the output pixel unit (OPU) which transfers the pixel to the next processor of the array, the microcoded control unit (MCU) and the microprocessor interface unit (MIU) which controls all interactions between the processor and the standard microprocessor.

Only four clock cycles are needed by the processor to perform all the operations associated with one pixel (parameters computing, velocity updating, boundary checking, ...).

The circuit will be fabricated in a 1.2 µm CMOS technology. The die size is approximatively 60 mm^2. The chip has been designed to have a clock frequency of 40 MHz which will allow to have a pixel frequency of up to 10 MHz.

6. CONCLUSION

An algorithm using the optical flow estimation and the smoothness constraint for multi-motion computing has been described. A systolic array allowing a real-time implementation in a standard video camera environment has been presented. This systolic architecture is based on a custom integrated circuit which integrates the full specifications of the elementary array processor.

7. REFERENCES

[1] D.G. Stavenga and R.D. Hardie, "Facets of Vision," Springer Berlin Heidelberg 1989.

[2] J.M. Pichon, C. Blanes and N. Franceschini, "Visual guidance of a mobile robot equipped with a network of self-motion sensors," *Mobile Robots IV* ,1989, SPIE vol. 1195, pp 44-53.

[3] J. Hutchinson, C. Koch, J. Luo and C. Mead, "Computing Motion Using Analog and Binary Resistive Networks," *Computer*, March 1988, pp 52-63.

[4] J. Wu, R. Brockett and K. Wohn, "A Contour-based Recovery of Image Flow : Iterative Method," *CVPR'89 - IEEE Conference on Computer Vision and Pattern Recognition*, 1989, San Diego, pp 124-129.

[5] J. Tanner and C. Mead, "Optical Motion Sensor", Analog VLSI and Neural Systems, Addison-Wesley, 1989.

[6] B. Horn and B. Schunck, "Determining Optical Flow," *Artificial Intelligence*, vol. 17,1981, pp 44-53.

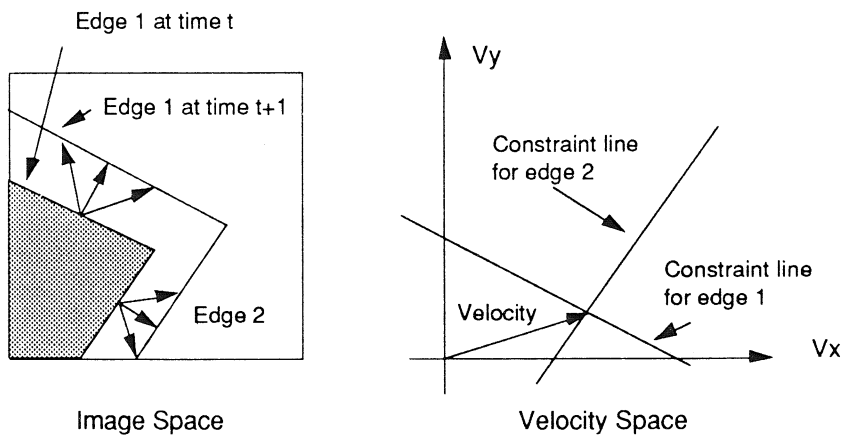

Figure 1 : the aperture problem and velocity determination
by the intersection of the constraint lines [5]

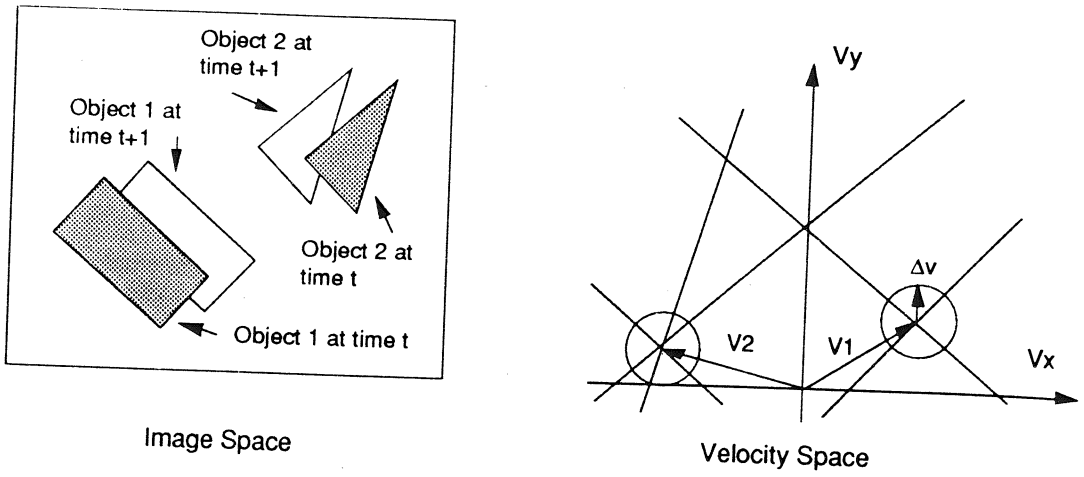

Figure 2 : multi-motion computing

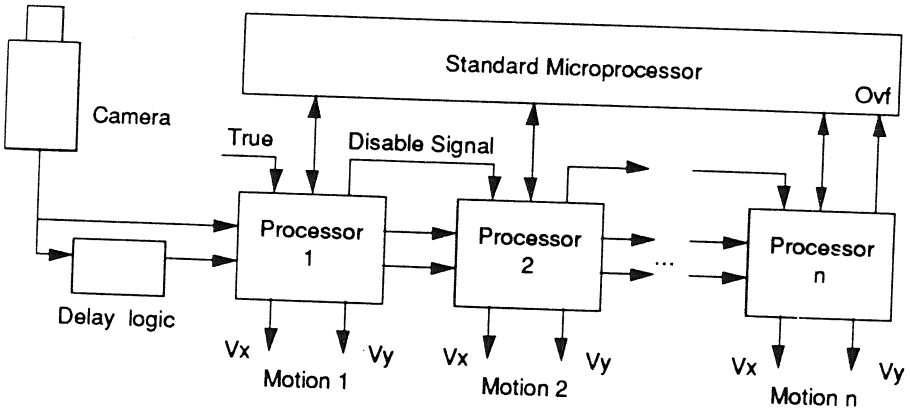

Figure 3 : systolic array architecture

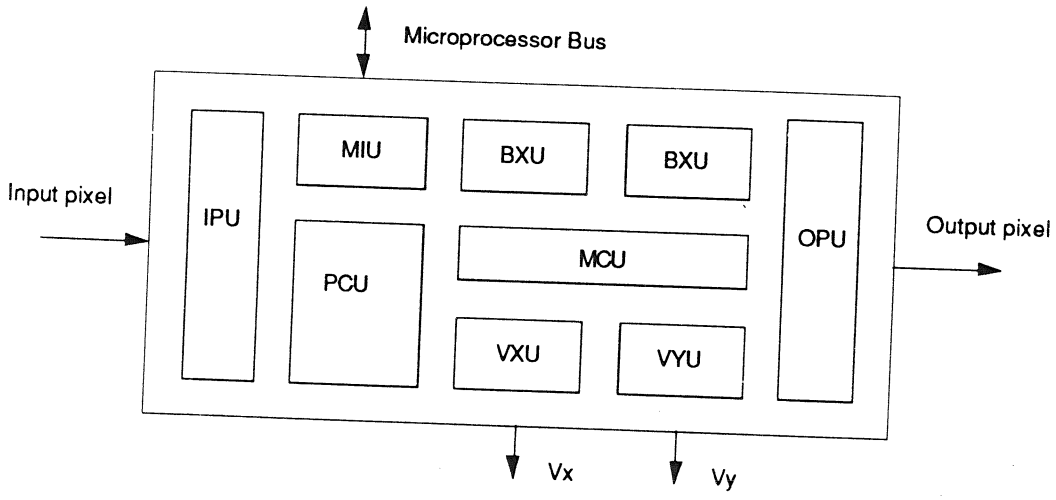

Figure 4 : circuit block diagram