INP 20
AN IMAGE NEIGHBORHOOD PROCESSOR FOR LARGE KERNELS.

D. David, T. Court, J.L. Jacquot, A. Pirson.
CEA/D.LETI
Av. des Martyrs, 85X
38 041 Grenoble cedex, France.

ABSTRACT

The neighborhood processor we present here has two very original features: first, it can handle neighborhood size up to 20 (and even more when cascaded) and second, it can handle linear operations (filtering and so on) as well as non linear operations (grey morphology, adaptative thresholding, ...). For example, it is able to perform 20x20 grey morphology, 9x9 "Canny-like" edge extraction or 20x20 local mean at HDTV rate (25 Mhz). The theoretical study of some particular properties of the implemented algorithms will be given, followed by the actual implementation of the processor.

INTRODUCTION

It is well-known that parallel architectures are necessary to process the considerable amount of data in low-level image transformations. A number of specialized processors have been designed for that purpose (HITACHI, LSI LOGIC, ...). But due to the great number of cells required to implement large kernels, the integration of corresponding systolic arrays is today difficult. However, we have identified large kernels to be helpful for many industrial inspection problems (e.g. large opening or closing, local mean and variance, high-quality edge extraction, ...). Moreover, another problem is the variety of the algorithms to be used. Therefore, designing a flexible enough image processing system results in large and expensive solutions. We began by the study of algorithms, which give best results on a broad range of images taken from real industrial problems (lack or presence of parts, surface aspect, microelectronic inspection, ...). Rather than designing a large circuit with n^2 cells (for nxn neighborhoods), we have opted for carefully studying some properties of these algorithms. Then, we have found a powerful architecture, based on those properties, and making possible to handle a broad range of algorithms on large kernels, while keeping the advantages of a small design (standard technology, low cost, easy test, high speed, ...).

PROPERTIES OF SOME DIGITAL FILTERS

Factorization: Let us consider, for example, the mean of m samples of a monodimensional (1-D) signal:

$$y_n = \sum_{k=0}^{m-1} x_{n-k}$$

Applying the Z-transform, we obtain:

$$Y = X \sum_{k=0}^{m-1} z^{-k} = X \frac{1 - z^{-m}}{1 - z^{-1}}$$

X and Y are respectively input and output of a linear system having as impulse response $(1-z^{-m})/(1-z^{-1})$. This corresponds to the following system:

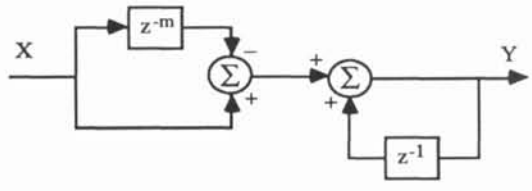


Figure 1. Block diagram of the mean filter.

As another example of factorization, the north SOBEL mask

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

corresponds to the following transformation:

$$Y = (1 + 2z^{-1} + z^{-2})(1 - z^{-2N})X$$

assuming that a classical raster-scan is used and that N is the size of this image. The corresponding block diagram is the following:

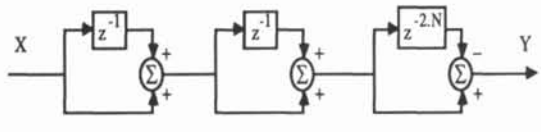
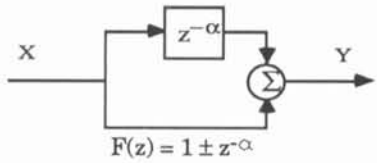
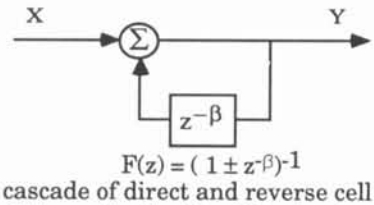


Figure 2.Diagram of the SOBEL edge extractor.

The filtering cells: We introduce now two kinds of cells.
- the direct cell:



- the reverse cell:



Any cascade of direct and reverse cell will permit to synthetize the following transform function:

$$H(z) = \frac{\prod_{i=1}^{N_f} (1 - p_i z^{-\alpha_i})}{\prod_{j=1}^{N_r} (1 - q_j z^{-\beta_j})}$$

(where $p_i, q_i \in \{1,-1\}$ and $\alpha, \beta \in N_0$).

A study of the stability domain of H(z) can be found in [1].

Synthetizing a linear transform: Most classical 3x3 masks can be implemented with the above cells. For larger masks, a software tool was developped for decomposing them into the right cascade of cells. If such a decomposition is not possible, the software tool can propose approximations by masks having z-transforms of the general form H(z). Another interesting possibility is to work directly on the frequency response of the linear filter to be implemented. The software tool proposes filters with frequency responses that best fit the desired frequency response. For example, a family of powerful edge extractors was constructed, based on CANNY's theory of edge detection. ([2], [3]).These extractors were introduced in [4] with a formal proof of their efficiency and a comparison with classical extractors. The 9x9 east (or north) mask, for example, requires only 12 cells (and no multipliers as it can be seen from the general form of H(z)). This mask and its z-transform are:

$(1\ 4\ 8\ 12\ 14\ 12\ 8\ 4\ 1)^T.(1\ 5\ 10\ 9\ 0\ -9\ -10\ -5\ -1)$

and

$$(1 + z^{-N})^4.(1 + z^{-2N})^2.(1 + z^{-1})^5.(1 - z^{-3})$$

It results in the following cascade of cells:

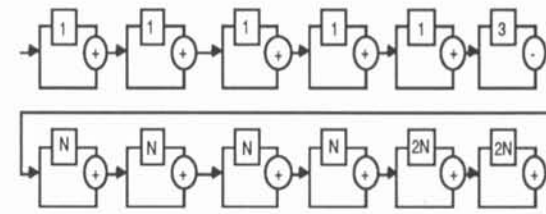


Figure 3. Block diagram of the 9x9 east edge extractor.

Non-linear filtering: In the case of grey-tone mathematical morphology, the following operation applies:

$y_{m,n} = \min \text{ (or } \max)_{k,l \in B} \{ x_{m+k,n+l} \}$

where B is the support of the structuring element. Defining D^k as a delay operator such that $D^k(x_n) = x_{n-k}$, the minimum of the last 4 samples (for example) can be written:

$y = \min\{ \min\{x_n, D^1(x_n)\}, D^2(\min\{x_n, D^1(x_n)\})\}$

This equation yields to the following block diagram:

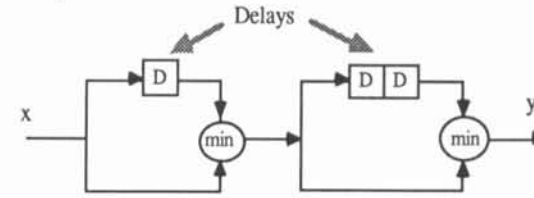


Figure 4. Block diagram of the minimum filter on a set of 4 samples.

It is easy to show that the extension to a filter of size m involves a cascade of M cells where :

$$M \leq E (\log_2 m) + 1$$

The basic cell for such filters is the following:

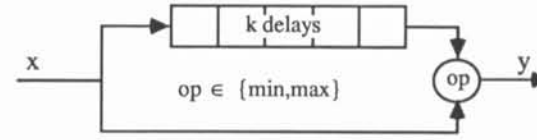


Figure 5. Basic cell for morphology.

We can see that this structure is very similar to that of the previously introduced direct and reverse cells.

Simulation: To test the validity of this approach on several algorithms (as well as to experiment the approximated linear filters), a simulation of the cascade of cells was written in OCCAM, and implemented on a Transputer

network. Some impressive results were obtained concerning the speed of operation at this stage of the work. (≈ 6 s. for one mask - north or east - of the 11x11 extractor, running on a 4-IMS T414-20 network (512x512 image) vs 90 s. for classical implementaion on a VAX).

IMPLEMENTATION

General structure: As shown above, the general structure involves a cascade of cells (direct, reverse, min/max). Each kind of cell is composed of two basic elements, the operator and the delay line.

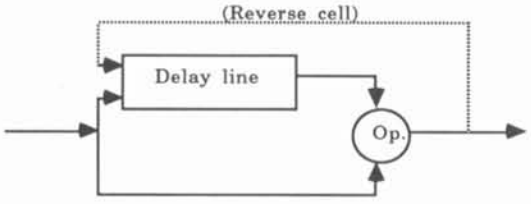


Figure 6. General structure of the basic cell.

1- *The operator:* The operator is a classical ALU, extended to perform min and max operations.

2-*The delay line:* It can be seen from the equation of H(z), that the delay lines can take values up to nxN, n being the size of the neighbourhood and N the size of the image (or the number of pixels per line). To integrate several cells on a sole device, we have to find room enough for several delay lines. This would be possible with today available technologies, but we preferred to choose another way, which yields much smaller (and more cost-effective) devices. Let us consider only the class of separable filters; H(z) becomes:

$$H(z) = H_x(z) \,.\, H_y(z^{-N})$$

The degree of H_x (or H_y) is equal to the size of the neighborhood. Using separability, a whole neighborhood processing will require two passes:
- the x-pass, where $H_x(z)$ is performed;
- the y-pass, where $H_y(z^{-N})$ is performed on the result of the first pass.

If the second pass is performed using a column per column raster-scan, the delay lines used for H_y can be divided per N.

Since most interesting kernels are separable, it is not an important limitation to restrict our chip to separable kernels (or separable structuring elements for mathematical morphology).

Conception: The chip was designed using ES2 / SOLO1000 software, which allows the implementation of devices up to ≈ 12000 gates. To cope with that limit, several compromises were considered:

1- *Number of cells versus length of delay lines:* Since the sum of the degrees of each factor of H(z) is equal to the size of the kernel, the total delay line of the device can be distributed on every cell: the greater the number of factors, the greater the number of cells, but the lower the degree of each factor, and the lower the length of each delay. Six cells were implemented, each one having a four clock-steps delay line; among these cells, k cells can be cascaded in order to form a new cell having a delay line of 4k.

2- *Dynamic-range of datas:* It is not necessary to have the same dynamic-range from input to output. However, it is a good idea to keep similar cells for conception and test convenience. Cells 1 to 5 have 12-bit data-paths, cell 6 has a 16-bit data-path. This is greater than most of actual convolvers, and gives the user a good security margin.

Final structure of the basic cell: All six cells are derived from the basic schema we give here, with small possible differences.

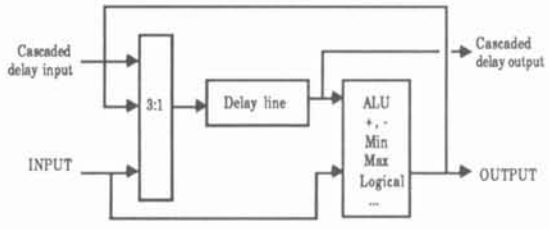


Figure 7. The basic cell.

Assembling the six cells: INP 20 includes six cells of the preceding type. To extend the range of possible applications and to get a more flexible circuit, the following features have been added:
- two 12-bit inputs on the first cell with appropriate selection circuitry,
- a special direct 12-bit path between one input of the chip and the fifth cell: it is possible to transform an image using cells 1 to 4, and to "mix" source and result images using cells 5 - 6 (+, -, and, min, ...),
- ability to process an additional tag-bit accompanying the input data. When an input pixel is marked, all output pixels depending on that input pixel (in relation with the size of neighborhood) will be marked too.

Programming: For each cell, programming the following parameters will cause different algorithms to be performed:
- type of operations for the operator,
- length of delay line,
- type of cell: reverse, direct, cascaded delay.

A software tool running on a standard computer gives these parameters, either automatically from the algorithms library (actually more than 150 items), or interactively by mean of a graphic tool representing the cells. The library includes parametrable edge extractors, mean filters, low-, band-, and high-pass filters, texture filters (Law, Chen,...), min-max filters, erosion, dilation ... For linear filters, the DFT of the programmed mask can be displayed.

Furthermore, the program takes into account the problem of the order of the cells. Given the equation H(z) of the desired algorithm, each factor of H(z) will be assigned to one of the six cells: due to commutativity, the order of the factors - and therefore the order of the cells - is theoretically of no importance. Nevertheless, the following problem could occur at the output of a reverse cell: the dynamic range of data at this output could increase as long as pixels are being processed (the reverse cell can act as an accumulator). Having in mind that our cascade of cells synthetizes a **finite** impulse response filter, we can state that the output of the filter has a fixed dynamic range, whatever the size of the image may be. Therefore, it is always possible to place reverse cells at locations such that no problems of width of datas arises. The example of the mean filter (figure 1) illustrates the problem: it is obvious that placing the reverse cell before the direct cell will cause divergence of the cell, which is not the case with the given order.

Performances: All algorithms of the following list are performed at pixel-rate over than 15 MHz. (25 MHz typ.), therefore faster than video-rate:

- erosion,dilation on 12-bit grey images, size of structuring element up to 20x20,
- local mean (or variance) on neighborhoods up to 20x20,
- local extrema (min,max) (up to 20x20),
- all common 3x3 filters (SOBEL,...),
- texture extractors : local statistics, LAW's masks, bandpass filters,
- high performance edge extractors (up to 11x11),
- edge thinning by selecting local maxima of map of gradients,
- image accumulation, image projection, all inter-image operations,
- possible combination of above algorithms.

The chip was implanted with E-BEAM technology (2µm CMOS). The number of gates is around 12000 and the surface is 80 mm^2. The package is a 84-pin LCC. Power dissipation at 25 Mhz is less than 2 watts.

CONCLUSION

Considering a broad range of algorithms used in industrial visual inspection, we found out that they share some common properties, making attractive their implementation. Moreover, the same structure applies for most of them, by changing only a few parameters. Therefore, INP 20 has been mainly designed to solve a class of algorithms but most of commonly used algorithms belongs to that class. Moreover, the chip is interesting for its flexibility and its ability to handle large kernels. The simplicity of the basic cell results in high-speed performance. INP 20 is already able to process 20x20 kernels faster than numerical TV-rate (25 Mhz). Future versions are planned, which will benefit from more advanced technologies: they will include longer delay-lines which will permit to handle some non-separable filters (e.g. the rolling-ball algorithm,...) on even larger kernels.

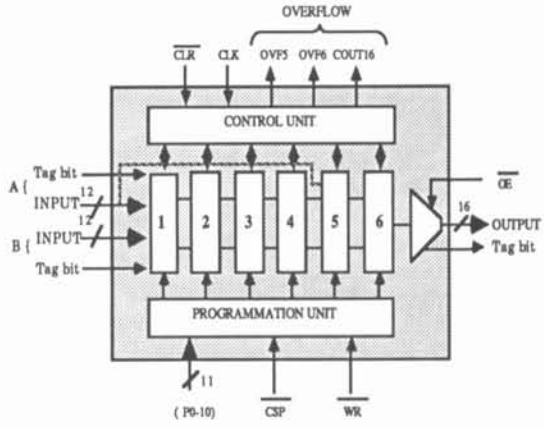


Figure 8. INP 20 block diagram.

REFERENCES

[1] A. PIRSON et al.: A highly efficient method for synthesizing some digital filters. Proceedings of EUSIPCO 88. Grenoble, France.

[2] J.F. CANNY: Finding edges and lines in images. MIT Artificial intelligence lab. TR-720, 1983.

[3] MARR & HILDRETH: Theory of edge detection. Proceedings of Royal Society of London, B. Vol. 207, 1980. pp. 187-217.

[4] J.L. JACQUOT et al.: Une classe d'extracteurs de contour performants adaptés à une implémentation matérielle fonctionnant à cadence vidéo. 2nd. workshop TIPI. CNRS. April 88. AUSSOIS, FRANCE.(A class of powerful edge detectors well suited for video rate implementation).