MULTISCALE EDGES DETECTION ALGORITHM IMPLEMENTATION USING FPGA DEVICES

M.Paindavoine, Sarifuddin ∗, C.Milan, J.C.Grapin
Laboratoire LIESIB − Universit de Bourgogne
6, Bd. Gabriel 21000 Dijon − France
e_mail paindav@satie.u − bourgogne.fr
∗ Electronic and Computer Lab. Gunadarma
jl. Margonda raya Pondok − cina − Depok
Jakarta − Indonesia

ABSTRACT

One of the way to extract edges uses the fast wavelet transform algorithm. This technique allows the detection of multiscale edge and is used to detect all the details which are in a picture by modifying the scale. The real time application for edge detection involves the implementation of the algorithm on an integrated circuit like an FPGA and the development of an appropriated board. This article deals about the implementation of a wavelet transform algorithm onto a FPGA and the development of an electronic board to detect multiscale edges.

1 INTRODUCTION

The image processing using the wavelet transform can be applied to the multiscale edges detection. In this way Mallat [1] introduces a fast wavelet algorithm using spline functions. We use the Mallat algorithm but using the generalised Deriche filter [2][3] and this to take advantages of the good Signal-to-Noise Ration performances of this filters.

With this algorithm, a four scale edges detection needs 32 filters third-order recursive filters (for a scale we need two detection filters and two smoothing filters).

We present in this paper a real time parallel implementation of our algorithm using FPGA devices.

2 FILTERS AND ALGORITHM DESCRIPTION

We can define a wavelet as a function $\psi_s(x)$ satisfying the condition $\int_{-\infty}^{+\infty} \psi_s(x)dx = 0$ where s represents the scale parameter. We can also introduce a scaling smoothing function $\phi_s(x)$ satisfying the condition that $\int_{-\infty}^{+\infty} \psi_s(x)dx = 1$. These wavelet and smoothing 1-D functions are realised using the generalised Deriche filters [3] :

$$\psi_s(x) = \{ {}^{C_s.(k_0 sxe^{msx} + e^{msx} - e^{sx})x \preceq 0}_{C_s.(k_0 sxe^{-msx} - e^{-msx} + e^{-sx})x \succ 0} \quad (1)$$

$$\phi_s(x) = A_s.[(k_0 ms^2|x| - k_0 s + ms)e^{-ms|x|} - ms^2 e^{-s|x|}] \quad (2)$$

with k_0 and m which are the adjustable parameters for the good detection, good localization and one response to one edge. These filters give an optimum Noise-to-Signal Ratio (NSR) and localization for $k_0 = -0,564$ and $m = 0,215\ \forall s$.

In the 2-D signals case, we can consider separable filters in x and y directions and in consequence we can define two masks x and y : $M^x = \psi_s(x)\phi_s(y)$ and $M^y = \phi_s(x)\psi_s(y)$.

For fast numerical implementations, the scale parameter is made to vary along a dyadic sequence $\{s = 2^j | j \in Z\}$. A dyadic wavelet transform of an image $f(x,y)$ is described by the convolution between an image and the two masks x and y. This convolution is shown by the following relations:
the X wavelet component

$$W^x_{2^j} f(x,y) = f(x,y) * (\psi_{2^j}(x), \phi_{2^j}(y)), \quad (3)$$

the Y wavelet component

$$W^y_{2^j} f(x,y) = f(x,y) * (\phi_{2^j}(x), \psi_{2^j}(y)), \quad (4)$$

and the vector gradient

$$W_{2^j} f(x,y) = \sqrt{|W^x_{2^j} f(x,y)|^2 + |W^y_{2^j} f(x,y)|^2}. \quad (5)$$

The X and Y wavelet transforms correspond respectively to the X and Y gradients of the image and the vector gradient represents the edge of image. Then we can use the wavelet transform for multiscale edge detection by the scale j modification.
The following algorithm represents a dyadic wavelet transform for multiscale edges detection.

for $j = 0$ and $j > J$
$W^x_{2^j} f(m,n) = (f(m,n) * \Psi_{2^j}(z)) * \Phi_{2^j}(z)$
$W^y_{2^j} f(m,n) = (f(m,n) * \Phi_{2^j}(z)) * \Psi_{2^j}(z)$
$W_{2^j} f(m,n) = \sqrt{|W^x_{2^j} f(m,n)|^2 + |W^y_{2^j} f(m,n)|^2}$
end for.
with $\Psi_{2^j}(z)$ and $\Phi_{2^j}(z)$ which are the numerical filters of the $\psi_{2^j}(x)$ and $\phi_{2^j}(x)$ filters. Their Z transform derives

two recursive filters moving in opposite directions :

$$\Psi^{+}_{2^j}(z) = \frac{a_1 Z^{-1} + a_2 Z^{-2}}{(1-\gamma Z^{-1})^2(1-\beta Z^{-1})} \quad (6)$$

$$\Psi^{-}_{2^j}(z) = \frac{a_1 Z + a_2 Z^{2}}{(1-\gamma Z)^2(1-\beta Z)}, \quad (7)$$

$$\Phi^{+}_{2^j}(z) = \frac{c_0 + c_1 Z^{-1} + c_2 Z^{-2}}{(1-\gamma Z^{-1})^2(1-\beta Z^{-1})} \quad (8)$$

$$\Phi^{-}_{2^j}(z) = \frac{c_3 Z + c_4 Z^{1} + c_5 Z^{2}}{(1-\gamma Z)^2(1-\beta Z)}. \quad (9)$$

with $a_1, a_2, c_0, c_1, c_2, c_3, c_4, c_5, \gamma$ and β are the numerical coefficients of the filters (table.1 and 2). These two Z-transforms correspond to two rational system transfer functions of stable third-order recursive filters from the left to the right (Ψ^+ and Φ^+) and from the right to the left (Ψ^- and Φ^-).

In particular, for exemple the output sequence $y(m)$ in response to $x(m)$ as input to a system, with impulse response $\psi(m)$, can be obtained recursively according to :

$y^{+}_{2^j}(m) = a_1.x(m-1) + a_2.x(m-2) + b_1.y^{+}_{2^j}(m-1) - b_2.y^{+}_{2^j}(m-2) + b_3.y^{+}_{2^j}(m-3)$

$y^{-}_{2^j}(M-m) = a_1.x(M-m-1) + a_2.x(M-m-2) + b_1.y^{-}_{2^j}(M-m-1) - b_2.y^{-}_{2^j}(M-m-2) + b_3.y^{-}_{2^j}(M-m-3)$

and

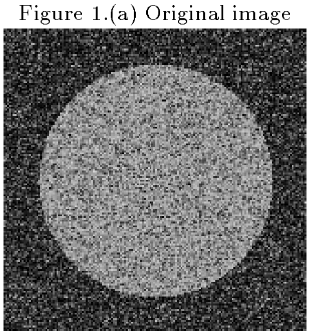
$$y_{2^j}(m) = y^{+}_{2^j}(m) + y^{-}_{2^j}(m) \quad (10)$$

for $m = 0, ...M$

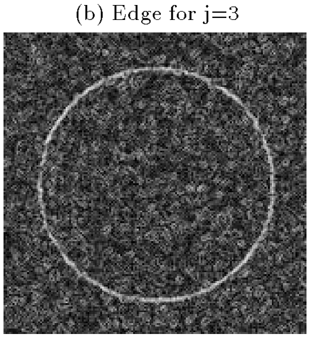
with $b_1 = 2.\gamma + \beta$, $b_2 = 2.\gamma.\beta + \gamma^2$, $b_3 = \gamma^2.\beta$ and M denoting the length of $x(m)$.

We denote as for each scale j, the algorithm (equations 3, 4 and 5) computes the edge of the original image. Then we can use this algorithm for parallely multiscale implementation. The following images represent the results of multiscale edges obtained by this algorithm.

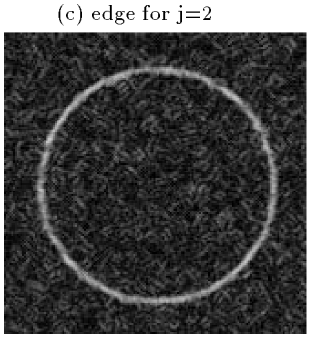
Figure 1.(a) Original image



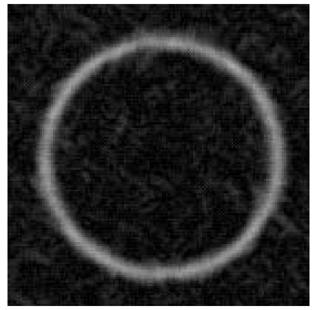
(b) Edge for j=3



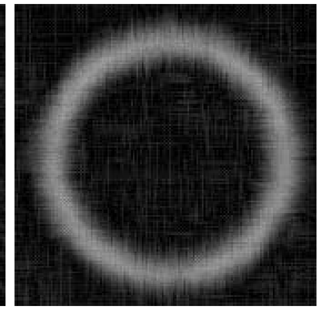
(c) edge for j=2



(d) Edge for j=1



(e) edge for j=0



The figures 1.(b), (c), (d) and (e) show the edges results of the noisy image (figure 1.(a)). We remark as if the scale j is large, the edges are very good localized but the noises are present. If the scale j is small, the noises are very good filtered and we have the edges very large. These edges results in the four different scales j obtained by the wavelet transform allow to detect all the detail informations present in a picture.

3 FILTER IMPLEMENTATION

In this section we present the implementation methods for the computation acceleration and the minimization of the circuitry surface. We propose a cascade transpose structure to implement our filter [4]. The equation (6) can be composed :

$$\Psi^{+}_{2^j}(z) = (a_1 + a_2 z^{-1}) z^{-1} \frac{1}{(1-\gamma z^{-1})} \frac{1}{(1-\gamma z^{-1})} \frac{1}{(1-\beta z^{-1})} \quad (11)$$

and its implementation structure is shown in figure (2). This structure gives an acceleration. The computation time is $T_{multiplication} + T_{addition}$ ($T_{multiplication} + 4.T_{addition}$ for classic structure implementation).

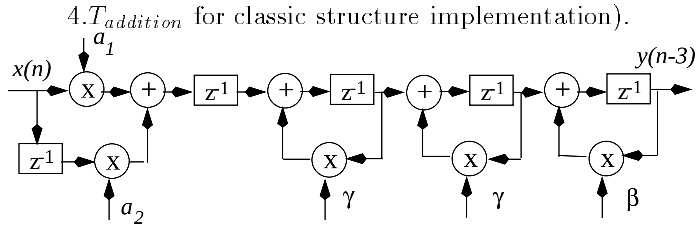


Figure (2). Cascade transpose implementation structure

For the minimization of the circuitry surface, we propose to implement the filters with the coefficients as constants. We choose the optimal case of the filters (equations 1 and 2 for $k_0 = -0.564, m = 0.215$) and we fix the scale $j = 0, 1, 2, 3$. In this case the numerical coefficients of the filters are interpreted as constants (tables 1 and 2) and this allows us to simplify the arithmetic operations in particular for the multiplication operations.

j	m	γ	β	a_1	a_2
0	0.21	0.8109375	0.3678794	0.0547381	−0.0320587
1	0.211	0.65625	0.1353352	0.1051479	−0.0368287
2	0.216	0.421875	0.0183156	0.3805066	−0.0519443
3	0.22	0.171875	0.0003355	0.7075226	−0.0022245

table 1.

j	c_0	c_1	c_2
0	0.0575068	-0.0586836	0.0131236
1	0.1155954	-0.063116	0.0045129
2	0.2365513	-0.0317419	-0.00194896
3	0.4719748	0.0335146	-0.0009251

c_3	c_4	c_5
0.0557411	-0.0590057	0.0139123
0.1042463	-0.0658027	0.0067374
0.1721808	-0.0477056	0.0007711
0.1959143	-0.0149224	0.0000047

table 2.

As example the multiplication betwen a coefficient a_1 (table 1 for $j = 2$) coded on 9 bits ($a_1 = 0.011000011_2$) and a pixel of image coded on 8 bits (d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0) is :

```
                              d7 d6 d5 d4 d3 d2 d1 d0
                           d7 d6 d5 d4 d3 d2 d1 d0
              d7 d6 d5 d4 d3 d2 d1 d0
           d7 d6 d5 d4 d3 d2 d1 d0
+ ______________________________________________________
  S7 S6 S5 S4 S3 S2 S1 S0 . S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8 S_9
```

This multiplication operation can be simplified by the following operation :

```
                    d7 d6 d5 d4 d3 d2 d1 d0
                 d7 d6 d5 d4 d3 d2 d1 d0
             + _____________________________
                 c9 c8 c7 c6 c5 c4 c3 c2 c1 c0
     c9 c8 c7 c6 c5 c4 c3 c2 c1 c0
+ ______________________________________________________
  S7 S6 S5 S4 S3 S2 S1 S0 . S_1 S_2 S_3 S_4 S_5 S_6 S_7 S_8 S_9
```

The expression of this operation is shown by figure (3). This multiplication operation uses 18 1-bit full and half adders and for classical method this multiplication uses 9x8=72 1-bit full and half adders. We have used this method to implement our filter and its allows to reduce between 70%-77% the circuitry surface.

4 ALGORITM IMPLEMENTATION

We choose to implement in a same FPGA device (XILINX 4005-PG156) the $\Psi^+_{2^j}$, $\Psi^-_{2^j}$, $\Phi^+_{2^j}$ and $\Phi^-_{2^j}$ filters. Thus for the four scale ($j = 0$, 1, 2 and 3) we use 8 FPGA devices (2 FPGA per scale j) as it is shown on the figure (4). This figure shows a parallel implementation algorithm because for one image in entry, we have four gradients of image parallely in the issue.

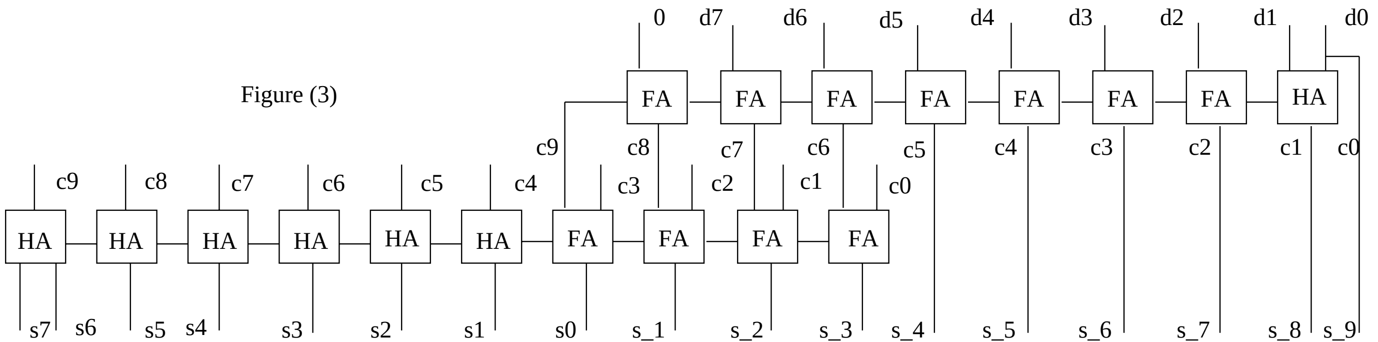


Figure (3)

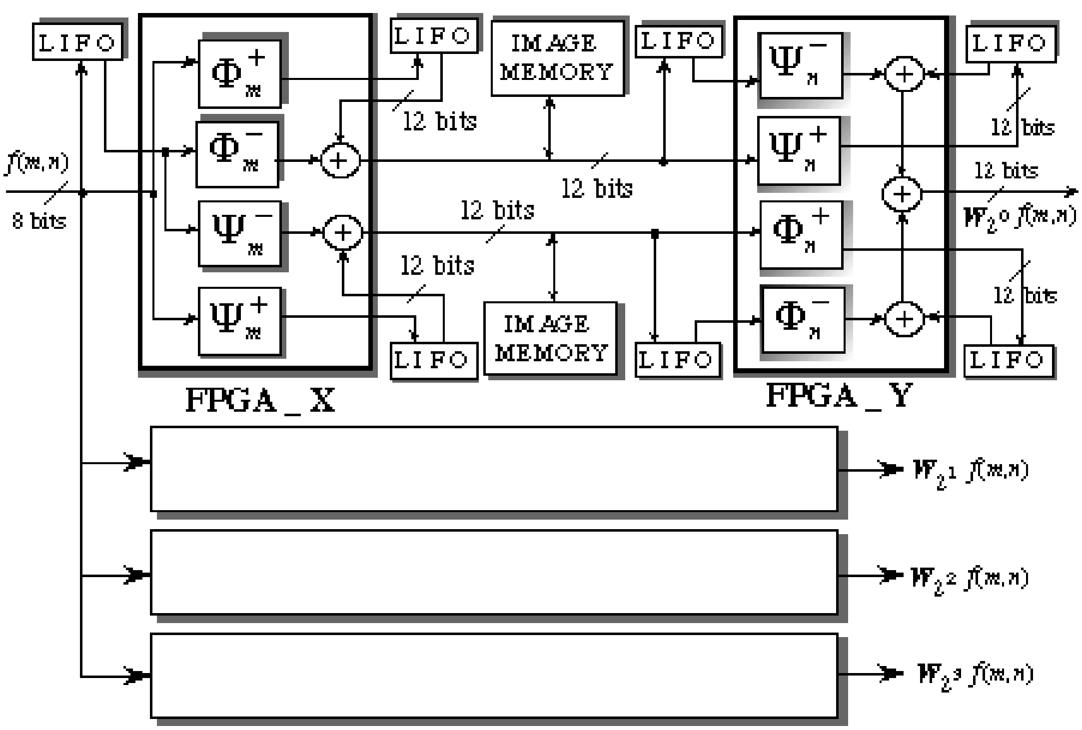


Figure (4)

CONCLUSION

We implemented the four filters $\Psi^{+}_{2^j}(z)$, $\Psi^{-}_{2^j}(z)$, $\Phi^{+}_{2^j(z)}$ and $\Phi^{-}_{2^j}(z)$, in the Xilinx FPGA 4005-PG156 device by using our methods. These filters use betwen 85%-90% of the FPGA surface.

We realized an image processing board in the goal to edges detection in real time (10 MHz per pixel for an image of 512 x 512 pixels) as it is shown on the figure (5).

The four scales edges detection allows to increase the quality of the detection in the case of blur and noisy images. Thus in consideration with the good edge detection quality in real time, we intend to use our system in industrial and medical applications.

BIBLIOGRAPHY

[1] S. Mallat and S. Zhong, "Characterization of Signal from Multiscale Edge", IEEE Trans. on PAMI, Vol. 14, No.7, pp.710-732, July 1992.

[2] R. Deriche, "Using Canny's Criteria to Derive a Recursively Implemented Optimal Edge Detector", International Journal of Computer Vision, Vol.1, No.2. pp. 167-187, Mai 1987.

[3] E. Bourennane, M. Paindavoine, F. Truchetet, "Amélioration du Filtre de Canny-Deriche pour la Détection des Contours Sous Forme de Rampe", Revue Traitement du Signal, Vol.10, no.4, 1993.

[4] Sarifuddin, "Implantation Sous Forme de Circuit Spécialisé d'un Algorithme de Détection de Contour Multi-échelles", Thèse, Université de Bourgogne, Juillet 1995.

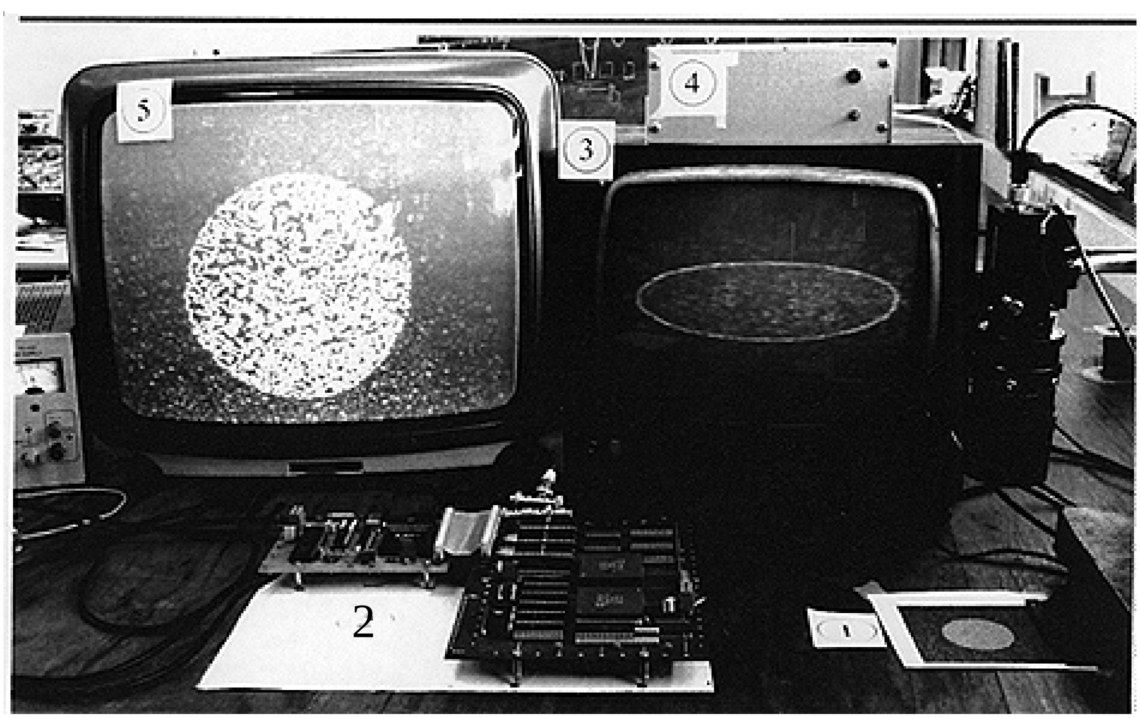


Figure (5). Real time multiscale edges detection system developed. 1. The original ilage, 2. Our real time image processing board, 3. The result of the edges by our board system, 4. The real time of Roberts gradient system box, The result of the edges by Roberts gradient system.