Implementation in an FPGA circuit of Edge detection algorithm based on the Discrete Wavelet Transforms

Implementation in an FPGA circuit of Edge detection algorithm based on the Discrete Wavelet Transforms

Issam Bouganssa[1], Mohamed Sbihi[1] and Mounia Zaim[1]

[1] Laboratory of System Analysis, Information Processing and Integrate Management, High School of Technology SALE, Mohammed V University in Rabat MOROCCO.

issam.bouganssa@gmail.com

Abstract. The 2D Discrete Wavelet Transform (DWT) is a computationally intensive task that is usually implemented on specific architectures in many imaging systems in real time. In this paper, a high throughput edge or contour detection algorithm is proposed based on the discrete wavelet transform. A technique for applying the filters on the three directions (Horizontal, Vertical and Diagonal) of the image is used to present the maximum of the existing contours. The proposed architectures were designed in VHDL and mapped to a Xilinx Sparten6 FPGA. The results of the synthesis show that the proposed architecture has a low area cost and can operate up to 100 MHz, which can perform 2D wavelet analysis for a sequence of images while maintaining the flexibility of the system to support an adaptive algorithm.

1. Introduction

Detecting edges within an image is an important characteristic of the image-searching process depending on the content. Faced with a large number of edge detection techniques and filters, it may be difficult to choose the most suitable approach for a specific collection of images, especially that there is no optimal technique for all the cases [1]. Moreover, it is crucial to study the characteristics of the images as a whole (synthetic or real image, fuzzy or clear, noisy) and according to their components (zones of high frequency versus low frequency) and then determine the most optimal edge detection approach in the selected set of detectors.

Edge extraction is used in the majority of cases, segmentation, image search depending on the content, identifying specific objects in images, or reconstructing objects in three dimensions, for this kind of recognition or reconstruction application [2]. Extracting the Edges is therefore an indispensable pre-treatment of the image before implementing other treatments.

In this paper, we use the Haar discrete wavelet transform for edge extraction to isolate the edge of the rest of the image in the horizontal, vertical and diagonal direction [3]. The algorithm is implemented in a Xilinx Spartan6 FPGA device that provides good performance of the integrated circuit platform for research and development. Indeed, FPGA technology has become an alternative for the implementation of software algorithms.

The results implemented in FPGA are displayed continuously on a screen via a VGA controller. The image will be stored in a FPGA block memory. The chosen technology allows real-time processing of image sequences while maintaining the flexibility of the system to support an adaptive algorithm.

2. Principe of contours

2.1. Frequency definition of contours

If the image is considered as a 2D signal, it is possible to pass in the frequency domain (by Fourier transform or wavelet for example) [4]. In this case, a contour can be seen as representing the high frequencies of the signal.

2.2. Spatial or temporal definition of contours

A contour can be seen as an abrupt change in the intensity of the image. There are several types of variations. A very similar way to the one quoted above is to consider the contours as a difference on the color. The contour detection filters this based on:

- Derivative first of a contour: gradient, Derivative operators of the first order, search for a passage through a maximum, or extra local maxima.
- Second derivative of an outline: Laplacian, Derivative operators of the second order, search for a passage by zero.
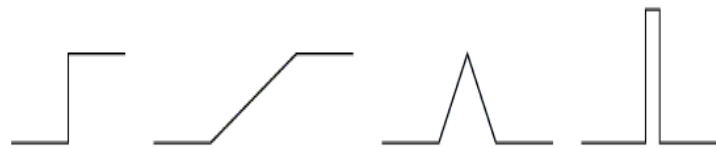

Figure 1. Contour profiles, Walking or ramp, roof, peak

2.3. Characteristics of contours

As well as the visual characteristics of color and texture, contours also have their peculiarities, to predict which detector will be the most effective. In order to estimate the efficiency of the detectors, we will refer to some errors encountered when detecting contours (Perfect contour Figure 2a):

- Omission of certain pixels on the contour to be detected. It is measured by counting the number of forgotten pixels with respect to the total number of pixels of the ideal contour (Figure 2b).
- Multiple responses by detecting multiple contours. It is measured by counting the number of ambiguous pixels compared to those that are not ambiguous (Figure 2d).
- Location: this error occurs when a pixel of an unambiguous ideal outline is not in the right place. It is measured by counting the total distance between the detected contour and the ideal contour. (Figure 2c).
- Sensitivity: This error is often related to noise and corresponds to false contours detected near the ideal contour. It is measured by counting the number of false contours and the total number of contours detected (figure 2e).

Figure 2. Examples of contour errors:
a: ideal contour,
b: Omission,
c: relocation,
d: multiple responses,
e: sensitivity.
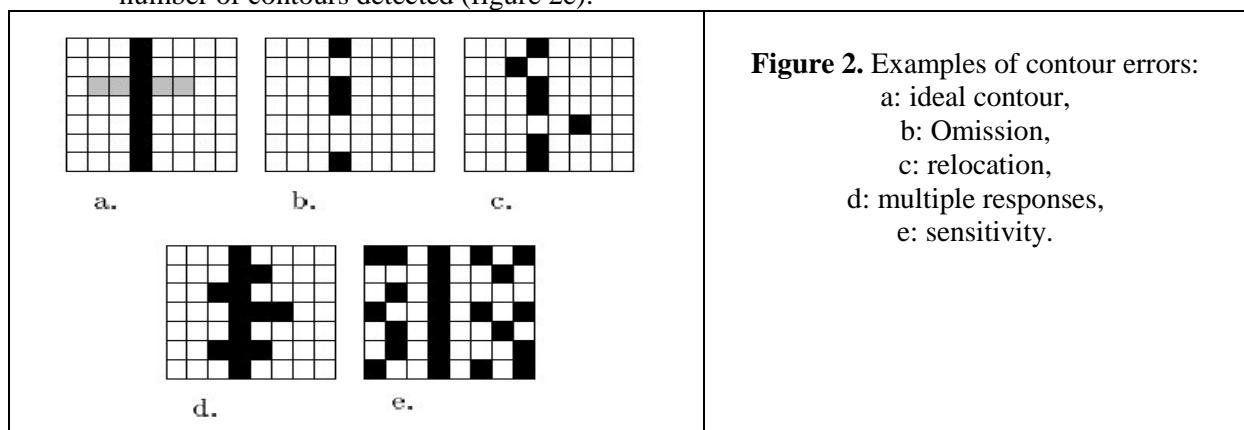
3. Wavelets

Most real-world signals are not stationary, and it is precisely in the evolution of their characteristics (statistics, frequency, temporal, spatial) that most of the information they contain is located. Voice signals and images are exemplary.

3.1. Fourier Transform

The Fourier analysis (equation 1) proposes a global signal approach, integrations are made from minus infinity to infinity [3,4], and any notion of temporal (or spatial) localization for images disappears in the space of Fourier.

$$T^{fourier} f(\omega) = \int_{+\infty}^{-\infty} f(t)e^{-j\omega t} \, dt \tag{1}$$

It is therefore necessary to find a compromise, a transformation which informs the frequency content while preserving the location in order to obtain a representation of the time / frequency signal.

3.2. Fourier transform with sliding window

The first solution that comes naturally to mind is to limit the domain of temporal (or spatial) integration by means of a "window" function that can be dragged to explore the signal[5]; we obtain the Fourier transform with sliding window (equation 2).

$$T^{f\,glisse} f(t, \omega) = \int_{+\infty}^{-\infty} f(s)g(s-t)e^{-j\omega t} \, ds \tag{2}$$

In this case, it is understood that the analysis is not ideal, because if a low temporal resolution is automatically linked to the detection of the low frequencies, the detection of the high frequency components of the signal can be done with a higher temporal resolution. The two resolutions must vary in the opposite direction, keeping a constant product for an energetically regular paving of the time-frequency space.

3.3. Wavelet transform

The Wavelet Transform Allows a rational use of the time-frequency space by the realization in all cases of the best possible compromise between the temporal and the frequency resolution [6].

This principle is specified in equation 3:

$$T^{ond} f(a, b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{+\infty} f(t)\psi(\frac{t-b}{a}) \, dt \tag{3}$$

In this expression, a is the scaling factor and b is the translation parameter. The variable a plays the role of the inverse of the frequency: More a is small less the wavelet is extended temporally, so the center frequency of its spectrum will bi higher.

This expression can also be interpreted as a projection of the signal on a family of analytic functions $\psi_{a,b}$ constructed from a "mother" function ψ according to equation 4:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) \tag{4}$$

It will be noted that the norm is preserved when the scaling factor is changed (equation 5):

$$\|\psi_{a,b}\|^2 = \int_{-\infty}^{+\infty} \frac{1}{a} \left| \psi\left(\frac{t-b}{a}\right) \right|^2 \, dt = cc \int_{-\infty}^{+\infty} \left| \psi(x) \right|^2 \, adx = \|\psi\|^2 \tag{5}$$

We can therefore: $T^{ond} f(a, b) = \langle f, \psi_{a,b} \rangle$

The spatial-temporal resolution is calculated in the same way as before: If the temporal "width" of ψ (the standard deviation) is taken as a unit: $\sigma = 1$ then the "width" of $\psi_{a,0}$ can be calculated with equation 6:

$$\sigma_t^2 = \int t^2 \left| \psi_{a,0}(t) \right|^2 dt = \int t^2 \frac{1}{a} \left| \psi\left(\frac{t}{a}\right) \right|^2 dt = \int a^2 x^2 \frac{1}{a} \langle \psi(x)\rangle^2 adx \tag{6}$$

Which gives $\sigma_t = a$ it is also possible to calculate the frequency occupancy of the wavelet by calculating the standard deviation for the Fourier transform $\psi_{a,01}$ of $\psi_{a,0}$ taking as unit the standard deviation of the Fourier transform of the mother wavelet ψ equation 7:

$$\sigma_\omega^2 = \int \omega^2 \left| \widehat{\psi_{a,0}}(\omega) \right|^2 d\omega = \int \omega^2 \frac{1}{a} \left| \widehat{a\psi}(a\omega) \right|^2 d\omega = \int \frac{\xi^2}{a^2} \frac{1}{a} \left| \widehat{a\psi}(\xi) \right|^2 \frac{d\xi}{a} \tag{7}$$

We find $\sigma = \frac{1}{a}$ So that the elementary block in the time-frequency space has a constant surface while the temporal resolution is proportional to a and the frequency resolution is inversely proportional to a as seen in figure 3.
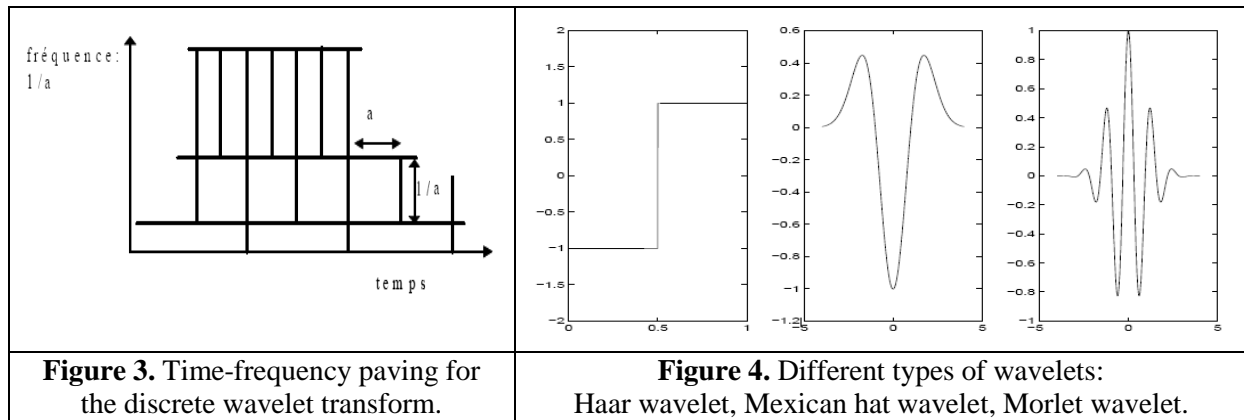
4. Principle of edge detection by wavelets

4.1. The types of wavelets use in 2D signal processing
The first wavelets used (apart from the **Haar** wavelet that we will study in this paper) have been the **Morlet** wavelet (a Gaussian modulated by a complex exponential) [7,8], **Daubechies** wavelets (compact support wavelets) and Wavelet of the **Mexican Hat** (the second derivative of a Gaussian).

Morlet:
$$\psi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} e^{-i\omega_o x} \qquad (8)$$

Mexican Hat:
$$\psi(x) = \frac{2}{\sqrt{3}} \pi^{-\frac{1}{4}} (1-x^2) e^{-\frac{x^2}{2}} \qquad (9)$$

| **Figure 3.** Time-frequency paving for the discrete wavelet transform. | **Figure 4.** Different types of wavelets: Haar wavelet, Mexican hat wavelet, Morlet wavelet. |

4.2. Principle of the Haar wavelet
The Haar wavelet is considered the first known wavelet. It is a piecewise constant function that will be denoted by H(x), and defined on the interval [0; 1] (or sometimes on [-1/2; 1/2]) which makes it the simplest wave to understand and implement moreover, its support is compact; it is well located in space. On the other hand, it has only one time zero and is discontinuous (equation 10 and figure 5):

$$H(x) = \begin{cases} 0 & si & x < 0 \\ 1 & si & 0 \le x < \frac{1}{2} \\ -1 & si & \frac{1}{2} \le x < 1 \\ 0 & si & 1 \le x \end{cases} \qquad (10)$$

This wavelet is very simple and is therefore easy to implement algorithmically [8]. Moreover, its support is compact; it is well located in space. On the other hand, it has only one time zero, and is discontinuous.

4.3. Algorithm for edge detection by the Haar wavelet transform
Wavelet transform algorithm we used is based on a 5×5 sliding window with a pitch of 2 pixels in the horizontal and vertical directions. The calculations are shown in equation (11), where d_0 and d_{01} are the outputs for each column of five pixels (p_{11} to p_{15}). Thus, for five columns [9], one obtains 5 d_0 and 5 d_{01}. Then, the two columns of d_0 and d_{01} are computed again using the same equations.
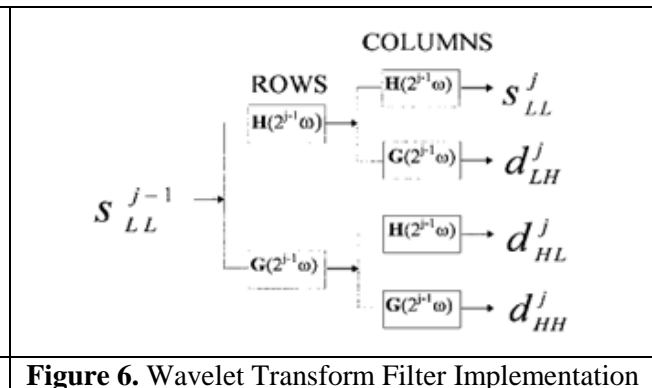
$$d_0 = (-1 \ 2 - 1) * \begin{pmatrix} p_{11} \\ p_{12} \\ p_{13} \end{pmatrix}; \quad d_1 = (-1 \ 2 - 1) * \begin{pmatrix} p_{13} \\ p_{14} \\ p_{15} \end{pmatrix}; \quad d_{01} = (d_0 + d_1) > 3 \qquad (11)$$

Therefore, each 5×5 window generates 4 output pixels, as is a sliding window in a step of 2 pixels at a time in both horizontal and vertical directions; we always get an output pixel per input pixel on average.

In the analysis phase of the two-dimensional wavelet transform (see Figure 6), each row of the input image is filtered separately by H and G [low-pass (H) and high-pass (G)]. The resulting pair of line-transformed images is also filtered in the column direction, giving four sub-band images at [J = 1]. The three images of "details" or simply called LH (low-high), HL (high-low) and HH (high-high) correspond to specific bands, not superimposed in the frequency domain. The "smooth" or LL (low-low) S_{LL} component is a low-pass filtered version of the original image, and is transmitted to the next level for further decomposition of the sub-band.
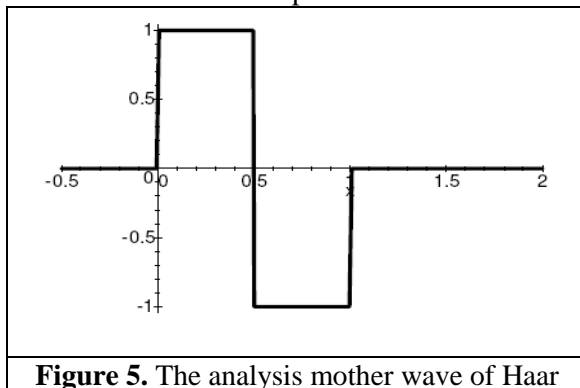
Figure 5. The analysis mother wave of Haar	**Figure 6.** Wavelet Transform Filter Implementation

4.4. Results of applications of the Haar DWT algorithms on a digital image

In this section, we present the results of the application of Haar's DWT contour detection method [10]. The table shows the application on different direction of the image, the horizontal detection of the contours, the vertical detection, and the diagonal detection.

Table 1. Application of Haar's DWT for different directions

I	 Original digital image	 Gray level image
II	 Application of Horizontal Haar DWT	 Application of Vertical Haar DWT
III	 Application of diagonal Haar DWT	 Application of operator Canny

From the results obtained, we can conclude that the detection of the contours by DWT of Haar gives results more precise and closer to the results obtained by the classical operator of Canny, especially in the direction of the contours.

5. Real-time image processing equipment

The processing of the image in real time requires a high computing power. For example, image standard.jpg with a dimension of 640 * 480 approximately 0.25 mega pixels per image, with a calculated power of 100 frames per second for a 25 MHz processor. The size of the image may be larger; the amount of processing required per pixel depends on the image processing algorithm.

Haar's DWT contour detection algorithm is a combination of over ten steps, horizontal detection, vertical and diagonal detection, and their convolution product that scans all pixels.

As high-resolution images become more frequent, processing requirements will increase, high-resolution images of standards generally have ten times more pixels per image, the calculation load is approximately ten times higher, and they require more DSP or a single DSP, very expensive high-end. In this scenario, FPGAs offer real-time alternative platform image processing. FPGA effectively supports high levels of parallel processing of data flow structures, which are important for the effective implementation of image processing algorithms.

6. Hardware implementation of the algorithm on a Xilinx FPGA circuit

Technical For the implementation of our algorithm, we used the Xilinx nexys3 platform (a board that consists of a Spartan-6 FPGA circuit plus I / O interface) and VGA monitor to display the results, the algorithm was developed on the Xilinx ISE interface and the various blocks are programmed in VHDL.

6.1 Block of "Memory" Program

The program block "Memory" that exists in the IP part of the Xilinx software is configured according to the size of the image, a square image for our example of 40.000 pixel which will store the image and each memory cell containing 8 bits in binary value from 0 to 255 (the encoded value for each RGB pixel).

6.2 Block of "Memory Read" program

Since the image is not necessarily the same size as the screen, the screen size and 640*480 and the image size 200*200. The program is necessary for the correct positioning of the image on the screen, so this is the reading part of the pixels stored in the memory.

6.3 Block program "VGA screen"

The VGA monitor is controlled by five 10-bit coded signals: red (3-bit), green (3-bit), blue (2-bit), horizontal (1-bit) synchronization and Vertical synchronization (1-bit). The three color signals, collectively known as the RGB signal, are used to control the color of a pixel at a location on the screen. In order to produce other colors, each color analog signal must be supplied with a voltage between 0.7 and 1.0 volts to vary the color intensities.

6.4 Block program "Synchronization"

The program allows you to synchronize the scanning of the pixels on the screen one horizontally and the other vertically. The signals are used to control the timing of the scanning speed. The horizontal synchronizing signal determines the scanning time of a line, while the vertical synchronizing signal determines the scanning time of the entire screen. By manipulating these signals, images are formed on the monitor screen.

6.5 Block program "algorithm"

This part of the program presents the contour detection algorithm used in this paper (the principle is detailed in section 4).

  After the conversion by applying the discrete wavelet transform, which is based on the FFT with a sliding window using the Haar wavelet, the pixels are always less than the total number of pixels [11].

  The upper limit ensures that operations according to the FIFO list can be completed during the real-time processing period of a frame; FIFOs are implemented using multiple built-in M4K memory blocks, each 4 kbps in size. The image of the edge detection result is stored in the RAM blocks embedded in the Spartan-6 device. These large integrated RAM blocks can be used to temporarily store the edge image when 2 bits are allocated to each pixel location.

  The following diagram shows the different blocks implemented on the software 'Xilinx ISE' [12,13]; software for programming the Xilinx FPGA circuits (figure 7).
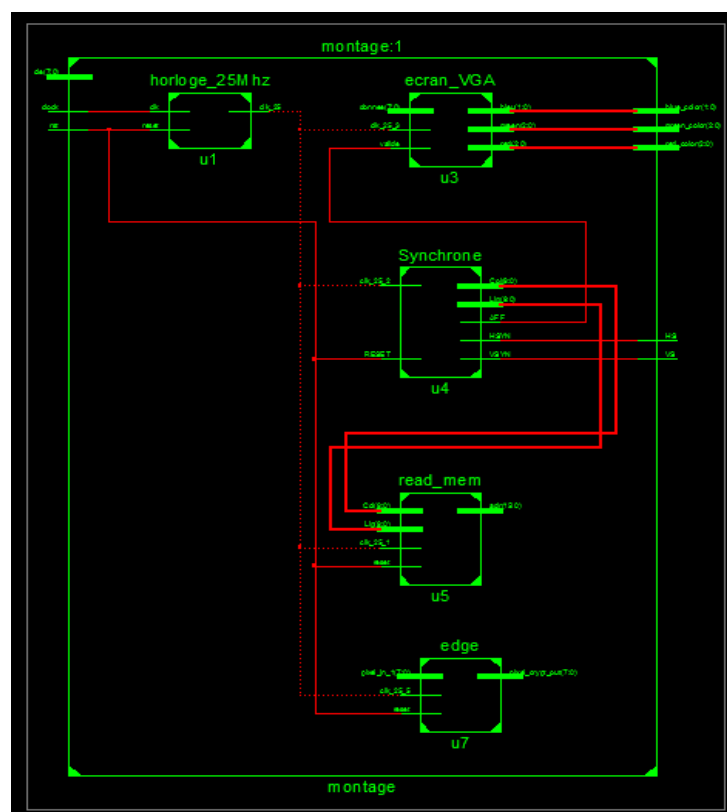
Figure 7. The blocks implemented on Xilinx ISE

## 7. Results and discussion

In Figure 7 We used the famous image in the field of image processing lenna.jpg for testing with a resolution of 200x200 pixels. The module is fully pipelined where a resulting pixel is computed at each clock cycle. With this rhythm and this clock frequency, it is possible to process more than 400 images at 200x200 pixels of resolutions.

Preliminary results show the FPGA design clocked at 100 MHz Spartan-6 targeting (figure 8).

  The design is scalable to handle high-resolution images while maintaining the clock frequency used to process standard resolution video at 30 frames per second, as well as high-speed computer vision applications that require more than 80 images per second.

  The use of high-resolution images requires a more powerful processor that exceeds the Spartan range, speaking of the Virtex range [14], high performance at frequencies above 500MHz. It is also possible to increase the frequency above 100MHz by introducing more pipeline stages to the detriment of an increased and optimal use of resources.
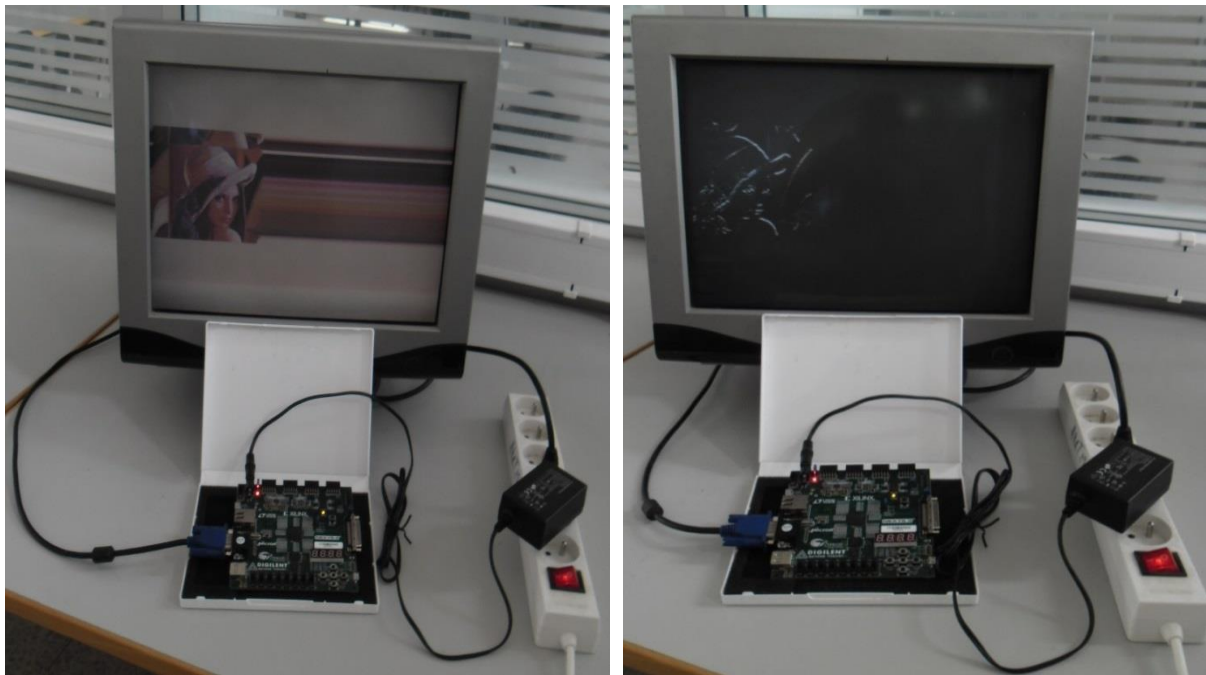
Figure 8. The image results before and after treatment.

8. Conclusions

This paper proposed the implementation of edge detection algorithms on FPGA architecture. The method presented is based on edge detection using the Haar discrete wavelet transform. The image is considered to be a 2D signal, we can pass in the frequency domain and in this case an edge can be seen as representing of the high frequencies of the signal.

The results obtained show the clear presence of all the edges that exist in the images (horizontal, vertical, and diagonal edges) [15,16]. The use of a spartan6 FPGA board demonstrates its ability to handle high-resolution VGA output images while maintaining the clock frequency used (100MHz) to process high-resolution image sequences at 30 frames per second as well as high-speed computer vision applications, which require more than 80 frames per second.

From the results obtained, we can conclude that the detection of the contours by DWT of Haar gives results more precise and closer to the results obtained by the classical operator of Canny, especially in the direction of the contours. For the improvement and performance of this work for the real-time processing of images, we propose as a prospect the use of a CMOS camera for the acquisition of real images, this camera will be connected to the FPGA card spartan6 Using a Pmod port connector, the results of the tightly processing images display permanently on a VGA screen.

9. References

[1]. T. Lindeberg, "*Edge Detection and Ridge detection with automatic scale selection*", International Journal of Computer Vision, Vol. **30**, No. 2, 117-154, 1998.

[2]. J. Malik, S. Belongie, T. Leung and J. Shi, "*Contour and texture analysis for image segmentation*", IJCV, vol.**43**, no.1, 2001.

[3]. Daubechies, Ten Lectures on Wavelets. Philadelphia, PA: SIAM, 1992.

[4]. Cohen, I. Daubechies, and J. C. Feaurean, "*Biorthogonal bases of compactly supported wavelets*", Commun. Pure Appl. Math., vol. **45**, 1990.

[5]. S. Mallat, W. Hwang, "*Singularity detection with wavelets*", IEEE Trans. Info. Theory, vol.**38**, pp. 617-643, 1992.

[6]. S. Mallat and S. Zhong, "*Characterization of signals from multiscale edges*", IEEE Trans. Pattern Anal. Machine Intell. vol. **14**, pp. 710–732, July 1992.

[7]. S. Mallat, "*A Wavelet Tour of Signal Processing. Orlando*", FL: Academic, 1999.

[8]. Grossman and J. Morlet, "*Decomposition of hardy functions into square integrable wavelets of constant shape",* SIAM J. Math Anal. vol. **15**, pp. 723-736, 1984.

[9]. M. Angelopoulou, K. Masselos, P. Cheung, and Y. Andreopoulos, "*A Comparison of 2-D Discrete Wavelet Transform Computation Schedules on FPGAs*", in IEEE International Conference on Field Programmable Technology, 2006, pp. 181-188.

[10]. J. Jyotheswar and S. Mahapatra "*Efficient FPGA implementation of DWT and modified SPIHT for lossless image compression*" Journal of Systems Architecture **53**, 369-378, 19 January 2007.

[11]. M.E. Angelopoulou, P.Y.K. Cheung, K. Masselos and Y. Andreopoulous "*Implementation and comparison of 5/3 Lifring 2D Discrete Wavelet Transform Computation Schedules on FPGAs*" Journal of Signal Systems **51**, 3-21, 2008.

[12]. http://store.digilentinc.com/fpga-programmable-log

[13]. http://www.xilinx.com/products/design-tools/ise-design-suite.html

[14]. S. Nazari, M. Amiri, K. Faez, and M. Amiri, "*Multiplier-less digital implementation of neuron–astrocyte signaling on FPGA*". Neurocomputing, 2015.

[15]. I. Bouganssa, M. Sbihi and M. Zaim "*Implementation of Edge Detection Digital Image Algorithm on a FPGA*" MATEC Web of Conferences. Vol. **75**, No 03003 Sept 2016.

[16]. I. Bouganssa, M. Sbihi and M. Zaim "*Implementation on a FPGA of Edge Detection Algorithm in Medical Image and Tumors Characterization*" DOI: **10**.1109/ICMCS.2016. 7905655. IEEE.Xplore 2016.