ACCELERATING STEREOSCOPIC VISION PROCESSING USING RECONFIGURABLE HARDWARE FOR EMBEDDED SYSTEMS

Jones Yudi Mori Alves da Silva, jonesyudi@unb.br
Carlos Humberto Llanos Quintero, llanos@unb.br
University of Brasilia, Faculty of Technology, Department of Mechanical Engineering, Automation and Control Group

Abstract. *Real-Time Machine Vision and Image Processing applications have high computational costs, mainly because of the low speed in common hardware/software architectures. In general, embedded systems implemented with microcontrollers and other Von Neumann-based hardware architectures cannot provide the throughput needed by these applications. Since their creation, the FPGAs (Field-Programmable Gate Arrays) have been used to accelerate data intensive computation, such as image processing and computer vision. This work presents an architecture specially developed for processing images in a real-time manner. The architecture allows the pixel stream to go through the processing elements, achieving a high throughput and speed-up performance. Basic filtering such as median, mean and convolution as well as some segmentation algorithms, e.g. erosion, dilation, opening and closing have been implemented in a parallel architecture for each camera of a stereo pair. Three well-known algorithms to extract depth information from stereo images are implemented in parallel, allowing to a comparison among them. The whole system is composed by two CMOS cameras, an low-cost FPGA and a LCD to visualize the processed images.*

Keywords*: Stereo Vision, FPGA, Embedded Systems, Image Processing, Computer Vision*

1. INTRODUCTION

One of the greatest challenges of our time is the development of systems that can reproduce, or at least approximate, the human capacity to feel and, especially, interpret the world. The human vision is not the best of the animal kingdom, but the speed with which our brain is capable of processing images, detecting patterns and extracting information is something extraordinary and extremely difficult to achieve with current technologies and mathematical models (Mori, 2010).

One of our more advanced perceptual abilities is our vision (composed by a pair of eyes), whose spatial arrangement allows the brain to get a sense of depth, allowing us to walk and perform tasks more accurately. This special provision of our eyes is known as stereo vision. A 2D image provides information on height and width only, whereas the 3D reconstruction allows to extract information concerning to depth. Stereo vision systems are becoming more available with the advent of 3D cinemas, domestic appliances, TV and video games. The Stereoscopic 3D reconstruction is an important process in the field of Computer Vision, and refers to the ability of inferring information about both the 3D structure and distances in a scene from a pair of stereo images (Pedrini, 2008).

The study of embedded systems is a field that currently requires a lot of attention taking into account the current technological moment. In recent years, products and technologies have been developed with the intention of aggregating great processing capabilities and intelligence, as well as robustness, flexibility and low power consumption, to the small systems. Embedded systems are hardware and software differentiated from those of common systems. They are generally used in applications with constraints on space, weight, energy, etc. and therefore they should have many of its functions optimized for providing reliability under diverse situations (Gokhale, 2005).

To accomplish image processing in real time some very expensive computing tasks need to be developed, since even small-size images are subject to several data mining operations such as filtering, color processing, thresholding, edge detection, etc. These operations require a high processing speed that most common systems cannot supply, thereby causing delays and failures in the applications (Kehtarnavaz, 2006).

The common hardware architecture a computer system has inherent delays arising from the so called “Von Neumann Bottleneck”. Such delays are enhanced whenever the amount of information to be processed is very high, such as in image processing systems. Common commercial cameras acquire images or videos, with a general rate of 30 frames per second. However, it is not difficult to find non-industrial applications with cameras that acquire more than 150 frames per second. Whenever these images can be stored for later analysis and processing, simpler systems equipped with hardware to capture and record at high-speeds can achieve the needs of the application. However, in real-time embedded systems, with memory, size, weight, speed and power restrictions, the presence of bottlenecks can lead to collapse (Eyre & Bier, 1998) (Patterson & Hennessy, 2005).

Several alternatives can be applied to solve the usual problems found in these architectures. This work focuses on the definition of dedicated architectures for processing images and videos with high-speed and flexibility, making use of algorithms specially adapted to be implemented in reconfigurable hardware. The use of FPGAs provides the flexibility

for developing the proposed system, and an important point is that they can implement the inherent parallelism of the algorithms used for image processing and computer vision in a natural way (Gokhale et.al., 2005).

The acceleration in performance through the implementation of algorithms in FPGAs is based on the fact that the algorithm to be implemented can be mapped directly in hardware, allowing for exploring its intrinsic parallelism. In the proposed architectures the data are entered on the hardware, being processed through a pipeline structure for achieving the desired results. By mapping an algorithm into hardware, the structures of loops, such as (for, repeat, while, etc.) are transposed in the pipeline, controlled by both the sequence of events and data counters (Hartenstein, 2006).

This paper presents a system for image processing in real time. A pipeline architecture allows the stream of pixels (coming from a camera) to pass through the processing elements, achieving a high performance. Basic filters such as median, mean and convolution, and some segmentation algorithms (e.g. erosion, dilation, opening and closing) were implemented in a low-cost system with two cameras, an FPGA and a LCD display for viewing the processed images, and their performance are shown and discussed. Additionally, an architecture for determining the disparities among pairs of stereo images is also proposed, allowing the system for performance prediction.

The rest of the paper is organized as follows: section 2 presents the theoretical basis and some assumptions made for architecture development. Section 3 shows some related works previously published in the literature. Section 4 describes the complete system and implementations carried out. Section 5 presents the conclusions and future works at this project, and Section 6 presents the references.

2. BACKGROUND

Computationally, a stereo system must solve two problems. The first one (that is known as correspondence problem) is to determine which item in the right camera image corresponds to which item in the left camera image (considering a system composed by two cameras only). The second problem consists on solving the stereo reconstruction. Our perception of the 3D world is due to the interpretation of the differences, called disparities, among two images acquired by our eyes. The group of all differences that are calculated from each point of the images is referred as disparity map, which in turn can be seen as an image. If the geometry of the stereo assembly is known, the disparity map can be converted into a 3D map view of the scene, a process known as 3D-Reconstruction (Trucco, 1998).

2.1. Epipolar Geometry

Both the assembly and calibration of stereo cameras must be carefully done so that the problem of 3D reconstruction can be simplified. Considering a system with two cameras (see Figure 1), where both cameras have the same focal length and are positioned with their optical axes in parallel. By simple trigonometry we can obtain the coordinates of point *P*, given by Equations 1, 2 and 3 (Trucco and Verri, 200X), (Perri, et.al., 2006).

$$z = \frac{T.f}{(x_1 - x_2)} \tag{1}$$

$$x = \frac{x_1}{f}(f - z) \tag{2}$$

$$y = \frac{y_1}{f}(f - z) \tag{3}$$

where T = distance between the optical axes, f = focal length of cameras, x_1 = x coordinate of P in an image, x_2 = x coordinate of P in image 2.

The value (x_1-x_2) is the disparity between two corresponding points in images 1 (left camera) and 2 (right camera). If the system is well calibrated and optical parameters are known, then values for each pixel disparity (disparity map) can be obtained by substituting this value in Equations 1, 2 and 3 to calculate the z, x and y coordinates of P in the real scene.

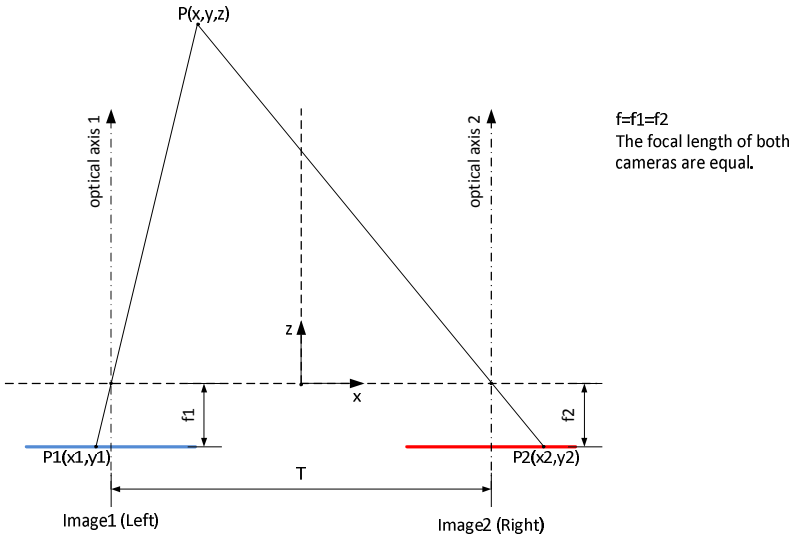

Figure 1 – Epipolar Geometry.

2.2. Image correspondences

There are basically two classes of algorithms to verify the match (or mismatch) between two images, one of them based on correlation and the other one based on image features. The correlation-based techniques should be applied to all points of the images, as long as techniques based on features establish the correspondence among some specific features from images (Trucco, 1998).

2.2.1 The correlation-based algorithms

In this type of method, windows of predefined size are compared with windows of similar size along the same line (epipolar line) in order to seek the best match. Various metrics for calculating the differences are proposed in the literature, the main being the Sum of Absolute Differences (SAD), the Cross-Correlation (CC), the Sum of Squared Differences (SSD) and Dynamic Programming (Trucco, 1998).

2.2.2 The feature-based algorithms

Methods based on features restrict the search for similarities in some regions as defined by characteristics previously identified in the images (corners, contours, lines, etc.). Each feature found in the first image is compared with all the features of the second image in order to identify which one presents the greater correspondence. Thus, one can obtain a list of differences among points that locate the corresponding features in both images (Truco, 1998).

3. RELATED WORKS

The algorithms to calculate the general disparities between the images (both based on correlation or in features) are quite expensive from computational point of view. The acceleration of these algorithms through its hardware implementation is proposed in several papers in the literature. Most related works use algorithms based on correlation as a metric to calculate similarity between images, and several architectures are proposed for implementing them on FPGAs.

In this context, Murphy et.al. 2007 performs the disparity calculation in images of 320x240 pixels at rates of 150 frames per second, using Census Transform. On the other hand, Georgoulas & Andreadis 2008 develop an extension of SAD using fuzzy logic to calculate the gap, reaching a throughput of 1713 frames per second, with resolution of 320x240 pixels.

Banz et.al. 2010 generates disparity maps using a technique known as Semi-Global Matching (SGM), being less sensitive to noise and lighting conditions. In this case, with low clock (39MHz) and VGA resolution (640x480 pixels), the system achieves a throughput of 30 frames per second. Kalomiris & Lygouras 2008, working at a frequency of 100MHz, calculate the differences for images of 320x240 pixels at a rate of 25 MPixel per second.

The system shown by Perri et.al. 2006 uses the SAD as a measure of similarity among regions, working at 286MHz on 512x512 pixel images with a rate of 25.6 frames per second. In another work, Kalomiris & Lygouras 2009 compare

two different techniques (based on SAD and Dynamic Programming) with images of 640x480 pixels, reaching rates of 162 and 81 frames per second, respectively.

Most of the previous works only have implemented architectures for disparity computations, by analyzing one unique algorithm for this task. Only a few of them show a system with real-time capture and processing. It's well known that the most intensive part of the whole 3D reconstruction process (capture, pre-processing, and disparities computation) is the disparities determination, so many previous work neglected the pre-processing tasks. In our proposal, the system takes advantage of the possibility of parallel acquisition and processing of images from two cameras running all the pre-processing operations in parallel on the two images. And then, with the two pre-processed images, the architecture implements three disparity metrics in parallel (SAD, SSD, Correlation), achieving a real-time performance. Because of this parallelism it's proposed to the future the analysis among these techniques, in order to know which of them can offer better results.

4. THE OVERALL SYSTEM

The hardware platform used in this work consists of the elements shown in Figure 2: (a) two cameras (Terasic, 2011) assembled side by side, with its focal axis parallel, (b) a development kit with Altera FPGA Cyclone II (Terasic, 2011) and (c) an LCD for viewing images (Terasic, 2011).

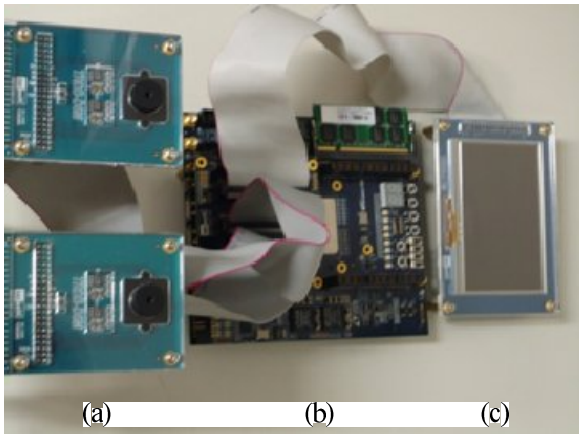

Figure 2 – Hardware Platform: (a) two cameras, (b) development kit. Kit, (c) LCD.

The proposed acquisition and image processing system is depicted in Figure 3. The architecture is divided into several blocks: *image acquisition*, *preprocessing* and *calculation of disparities*. As shown in Figure 3, there are two pre-processing units, running in parallel, accelerating the overall system performance.

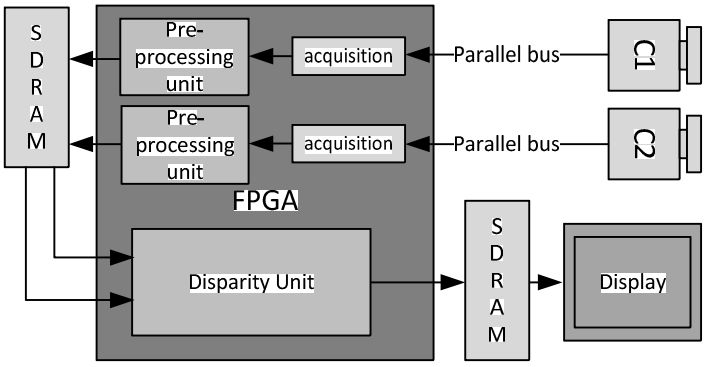

Figure 3 – Hardware Platform: *image acquisition*, *preprocessing* and *calculation of disparities*.

4.1. The Pre-Processing Unit

Figure 4 (a) shows in detail the implemented architecture for pre-processing task of the images. Each camera provides the data (pixel by pixel) as a stream, which is synchronized by a single clock. The processing architecture

shown in Figure 4 (a) was designed in order to implement a synchronous pipeline, generating a processed pixel every clock cycle, after an initial latency period. Table 1 shows the performance results of the processing blocks of Figure 4 (a).

Table 1. Performance results for the blocks of Pre-Processing Unit.

Block	Processing Delay (clock cycles)
Image Acquisition	6
RGB2Gray	2
Neighborhood Loader	1603
Median Filter	9
Neighborhood Loader	1603
Convolution	2
Absolute Sum	1
Neighborhood Loader	1603
Gray-Scale Dilation	9
Gray-Scale Erosion	9
Total	4847

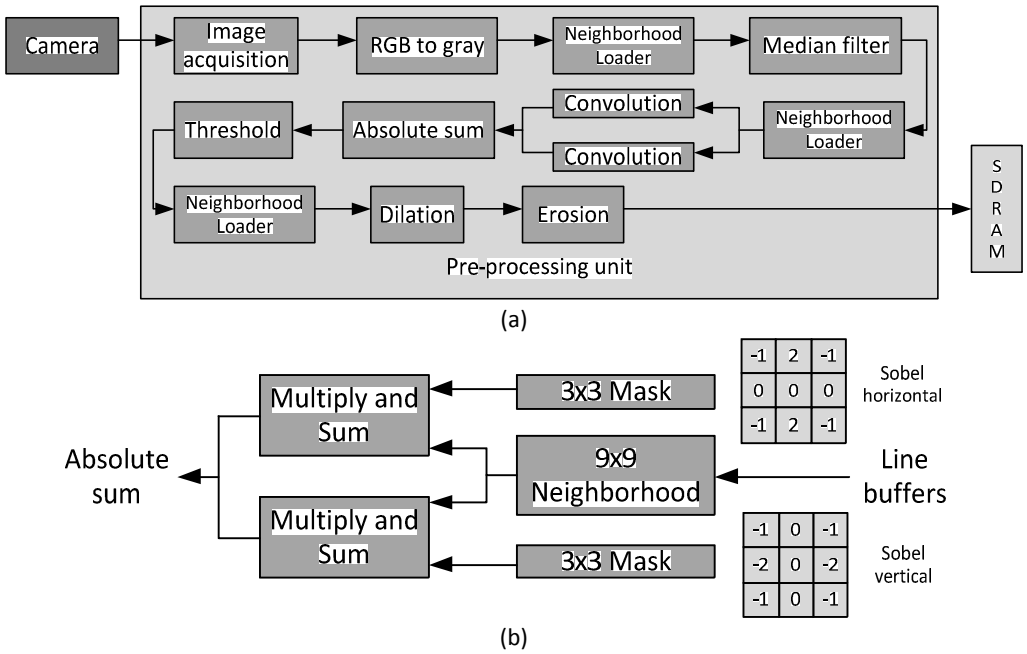

Figure 4 – (a) Pre-processing unit details. (b) Sobel filtering by two parallel convolutions.

Note also that in Figure 4 (a) there are two convolution operations occurring in parallel. Figure 4 (b) details this part of the processing, showing that operations that can work independently (for example, vertical and horizontal Sobel filters) are implemented in parallel for increasing system throughput. The architecture for *pre-processing* is capable to run at a maximum frequency of 54MHz and the same is capable to process an image of 800x480 pixels in 388,847 clock cycles, i.e. a rate of about 138 images per second.

4.2. Disparity Unit

The method chosen to calculate the disparity is based on correlation, and we proposed three architectures similar in structure, changing only the metric used to determine the similarities (disparities). The metrics selected were: (a) SAD, SSD and (b) Correlation. By observing their behavior, one can observe that all three metrics have in common the fact that they work over image windows. For each window in the left image, one must walk through the same line in the right image, calculating the degree of similarity, as shown in Figure 5.

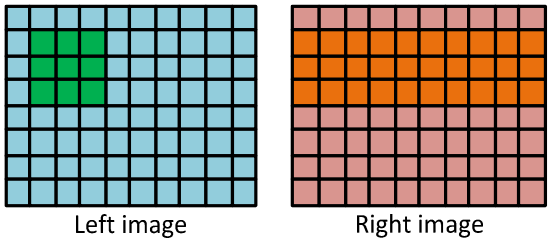

Figure 5 – Windowing process.

The blocks named *neighborhood loader* (see Figure 5) provide the windows of right and left images to blocks SAD, SSD and *correlation* at the same time, thereby the same architecture for making the neighborhoods analyzed is used for all three metrics in a concurrent way.

The SDRAM memory shown in Figure 6 is filled with the two images (right and left) at a rate of 1 pixel per clock cycle. The model used for SDRAM memory has two write-ports and two read-ports for simultaneous operations, i.e., the two images are stored in parallel, leading to 38,400 clock cycles for storing them.

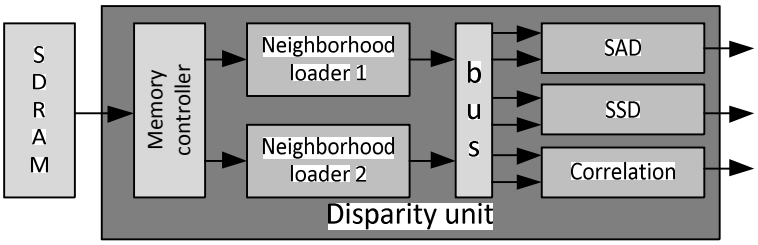

Figure 6 – Disparity unit diagram.

The process of hiring neighborhood for the three metrics of disparity occurs from simultaneous reading of the neighborhoods of each image by both blocks "Neighborhood Loader" and it has a delay shown in Table 1. Table 2 shows the number of clock cycles that each block of Figure 6 takes in its processing.

Table 2. Performance results for the blocks of Disparity Unit.

Block	Processing Delay (clock cycles)	Maximum Frequency (MHz)
Neighborhood Loader (1 e 2)	1603	235,07
SAD	2	420,17
SSD	18	122,33
Correlation	2	420,17
Total	SAD(1605) / SSD(1621) / Correlation(1605)	SAD(235,07) / SSD(122,33) / Correlation(235,7)

Like the pre-processing unit, the unit for calculation of disparities is also designed as a pipelined structure, so that after an initial delay (shown in Table 2) at each clock cycle a new value of disparity is available at the output of metrics blocks. The maximum operating frequency used is 300MHz (of the SDRAM), which implies that the maximum frequency of the *disparity unit* is limited by its constituent blocks.

Summing the delays of pre-processing blocks we generate the data in Table 3, which shows the total delay for each measure of disparity.

Table 3. Total delay to each disparity metric.

Disparity metric	Total Delay (clock cycles)
SAD	4847+1605=6452
SSD	4847+1621=6468
Correlation	4847+1605=6452

Considering the amount of pixels in each image in this design (800x480 pixels), and knowing the maximum frequency values for each block, we can estimate the throughput of each metric, in terms of frames per second.

As explained in section 2, every neighborhood of an image is compared with a range of image 2, along the same line. Fixing the search region with 40 pixels (disparity range), we have a total of 16,000 comparisons per line. Considering that the search is made for every three lines (due to the size of the neighborhood considered), the total amount of comparisons would be 2,560,000 per image.

After the initial delay of Table 3, at each clock cycle a calculation of disparity is completed. Table 4 shows the processing rate for each measure of disparity, considering the search region of 40 pixels.

Table 4. Processing rate to each disparity metric.

Disparity metric	Processing rate (frames per second)
SAD	30
SSD	14,8
Correlation	30

We can find at Table 4 that a common video frame rate (30 fps) can be realized by our architecture, showing that this implementation is suitable for general applications that require stereo vision at this rate.

5. CONCLUSIONS AND FUTURE WORK

This paper presented an architecture for image processing embedded in FPGAs designed in order to enhance the processing performance in stereo pairs of images. The system is mainly composed of two CMOS cameras and a FPGA Development Kit. The architecture is divided into two parts: (a) pre-processing and (b) calculation of disparity, with a high degree of parallelism of operations, allowing the differences to be calculated at rates up to 30 frames per second. This processing rate allows the use of the system in mobile robotic applications (mainly in visual servoing feedback) and some common video applications (pre-processing algorithms are present in the majority of image processing applications).

The pre-processing of images is performed independently and simultaneously for each camera, reducing the delays inherent in sequential processing found in common hardware platforms. The calculation of disparities is made in a dedicated architecture using three metrics simultaneously. The proposed system has low cost and low power consumption, a fundamental requirement in embedded applications.

The continuation of this project is intended to implement algorithms for calculating disparities based on features, which can offer better results. Some techniques will be implemented as detection of corners, regions segmentation, invariants, etc.. With the implementation of these techniques simultaneously, we can compare their performance according to the processing time and quality of results.

6. REFERENCES

Banz, C., Hesselbarth, S., Flatt, H., Blume, H., Pirsch, P., 2010. “Real-Time Stereo Vision System using Semi-Global Matching Disparity Estimation: Architecture and FPGA-Implementation”, Proceedings of 2010 International Conference on Embedded Computer Systems, Samos, Greece.

Banks, J., Corke, P., 2001. “Quantitative Evaluation of Matching and Validity Measures for Stereo Vision”. The International Journal of Robotics Research, <http://ijr.sagepub.com/content/20/7/512>.

Eyre, J., Bier, J., 1998. “DSP Processor hit the mainstream”, IEEE Computer, August 1998.

Georgoulas, C., Andreadis, I., 2008. “Real-Time Stereo Vision Techniques”, Proceedings of 16th IFIP/IEEE International Conference on Very Large Scale Integration, Rhodes, Greece.

Gokhale, M.B., 2005. “Reconfigurable Computing – Accelerating computation with Field-Programmable Gate Arrays”, 1st edition, Ed. Springer, 2005.

Hadjitheophanous, S., Ttofis, C, Georghiades, A.S., Theocharides, T., 2010. “Towards Hardware Stereoscopic 3D Reconstruction – A Real-Time FPGA Computation of the Disparity Map”, Proceeding of 2010 Design, Automation & Test in Europe, Dresden, Germany.

Hariyama, M., Yamashita, K., Kameyama, M., 2008. “FPGA Implementation of a Vehicle Detection Algorithm Using Three-Dimensional Information”,

Kalomiros, J., Lygouras, J., 2009. “Comparative Study of Local SAD and Dynamic Programming for Stereo Processing Using Dedicated Hardware”. Journal on Advances in Signal Processing, Vol. 2009, ID 914186, 18 pages.

Kalomiros, J.A., Lygouras, J., 2008. “Hardware implementation of a stereo co-processor in a medium-scale field programmable gate array”. IET Comuter 7 Digital techniques, Vol. 2, No. 5, pp. 336-346.

Mori, J.Y., Sánchez-Ferreira, C., Muñoz, D.M., Llanos, C.H., Berger, P., 2011, "An Unified Approach for Convolution-Based Image Filtering on Reconfigurable Systems", Proceedings of 2011 VII Southern Conference on Programmable Logic, Córdoba, Argentina, pp. 59-64.

Mori, J.Y., 2010. "Implementação de Técnicas de Processamento de Imagens no Domínio Espacial em Sistemas Reconfiguráveis", Master Thesis on Mechatronics Systems, University of Brasilia, Brasília, Brazil.

Murphy, C., Lindquist, D., Rynning, A.M., Cecil, T., Leavitt, S., Chang, M.L., Olin, F.W., 2007, "Low-Cost Stereo Vision on an FPGA", Proceedings of 2007 International Symposium on Field-Programmable Custom Computing Machines.

Patterson, D., Hennessy, J., 2005. "Computer Organization and Design – the Hardware/Software interface". 3rd edition, Ed. Elsevier, 2005.

Pedrini, H., 2008. "Análise de Imagens Digitais – Princípios, algoritmos e aplicações". 1st edition, Ed. Thomson, 2008.

Perri, S., Colonna, D., Zicari, P., Corsonello, P., 2006. "SAD-Based Stereo Matching Circuit for FPGAs", Proceedings of 2006 International Conference on Electronics, Circuits and Systems, Nice, France.

Terasic, 2011 – http://www.terasic.com.tw/ - accessed on March 2011.

Trucco, E., Verri, A., 1998. "Introductory techniques for 3-D computer vision". 1st edition, Ed. Prentice-Hall, 1998.

7. RESPONSIBILITY NOTICE

The author(s) are the only responsible for the printed material included in this paper.