14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006, copyright by EURASIP

FPGA ARCHITECTURE FOR OBJECT SEGMENTATION IN REAL TIME

Jozias Parente de Oliveira[*], *André Luiz Printes*[*], *Raimundo Carlos Silvério Freire*[**],
Elmar Uwe Kurt Melcher[**] , *and Ivan Sebastião de Souza e Silva*[***]

[*]Genius Institute of Technology, Av. Açaí 875, 69075-904, Manaus, Amazonas, Brasil. Phone: +55-092-36146580
Email: joliveira@genius.org.br and aprintes@genius.org.br

[**]Federal University of Campina Grande – Electrical Engineer Department, Rua Aprígio Veloso 882, 58109-970, Campina Grande, Paraíba, Brasil. Phone: +55-083-8803-6878. Email: freire@dee.ufcg.edu.br and elmar@dee.ufcg.edu.br

[***]Federal University of Pará - Electrical Engineer Department, Rua Augusto Corrêa 01, 66075-110, Belém, Pará, Brasil.
Email: ivan@ufpa.br. Phone: +55-091-8826-4791

ABSTRACT

Object segmentation from a video sequence is a function necessary for many applications of artificial vision systems such as: video surveillance, traffic monitoring, detection and tracking for video teleconferencing, video editing, etc. In this paper, we present an architecture for object segmentation, taking advantage of the data and logical parallel opportunities offered by a field programmable gate array (FPGA) architecture. At a clock rate of 40 MHz, the architecture can process 30 frames per second, where the image resolution is 240 x 120..

1. INTRODUCTION

Object segmentation is an essential part of information extraction in many computer vision applications including: video surveillance, traffic control and video coding. Most of the systems that detect and recognize objects in images perform the object extraction by subtracting the current image from a reference image to classify pixels belonging to the foreground and those of the background or reference image. In the last few years, many methods have been developed for object extraction using different techniques such as: block-based technique [1], statistical approach [2,3,4], Bayesian decision [5,6], mixture of gaussians [7]. Most of the existing algorithms have only been implemented in software and only a few in FPGA [8] and in VLSI [9].

An implementation in hardware is one approach to perform the object segmentation in real time. Horprasert et al [2] proposed a statistical nonparametric technique for background subtraction and shadow detection. They implemented this technique in a Pentium II – 400 Mhz using floating points operations to process 320 x 240 images at 27.6 ms per frame.

In this paper, we propose an FPGA architecture for object segmentation that can perform the scene analysis in real time, that is, 30 frames per second. We implemented the technique proposed by Horprasert et al. [2] using fixed points operations and a clock rate of 40 MHz. This paper is organized as follows: Section 2 presents FPGA implementation of the technique proposed by Horprasert et al. [2]. Sections 3 and 4 review the performance of the system. Section 5 draws some key concluding remarks on the work presented in this paper.

2. PROPOSED HARDWARE SET-UP

Horpraset et al. [2] consider the color constancy ability of human eyes and exploit the Lambertian hypothesis to consider color as a product of irradiance and reflectance. The basic steps of the algorithm are as follows:

First, background modeling constructs a reference image representing the background. The background is modeled statistically on a pixel-by-pixel basis. A pixel is modeled by the expected color value, the standard deviation of color value, the variation of brightness distortion, and the variation of chromaticity distortion. Second, a histogram of the normalized chromaticity distortion and a histogram of normalized brightness distortion are constructed. From the histograms, the appropriate thresholds are determined automatically according to the desired detection rate. Finally, subtraction operation or pixel classification classifies the type of a given pixel, i.e., the pixel is the part of background (including ordinary background and shaded background), or it is a moving object. A pixel in a current image is a moving foreground if the pixel has chromaticity different from the expected values in the background image.

Figures 1 to 3 show the block diagram of the architecture for object segmentation proposed in this paper. The Video Decoder block converts the analog video input to

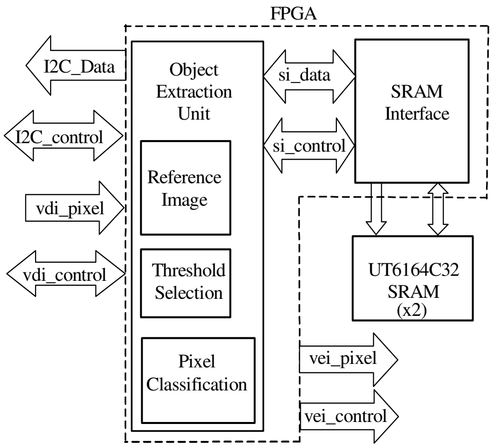


Figure 1: Image Acquisition Unit

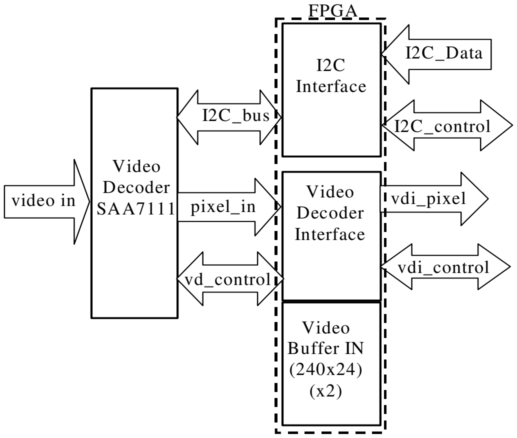


Figure 2: Image Segmentation Unit

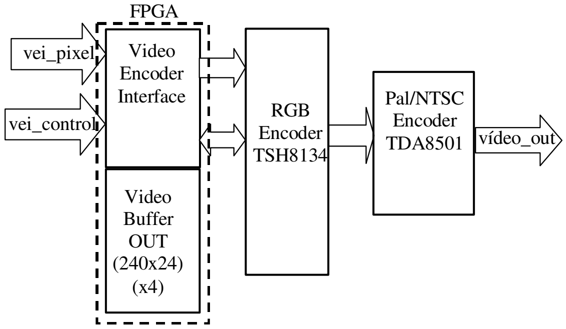


Figure 3: Image Exhibition Unit

digital RGB format. It provides 720 pixels per line and 480 lines per frame. The CMOS circuit SAA7111 from Philips Semiconductors is used for this application. The FPGA contains the following blocks: I2C Interface, Video Decoder Interface, Video Buffer IN, Object Extraction Unit, SRAM Interface, Video Encoder Interface and Video Buffer OUT. The Video Decoder Interface acquires the digitalized image from the Video Decoder and stores the pixels in the Video Buffer IN. In our implementation we choose 240 pixels per line and 120 lines per frame. As the Video Decoder provides 720 pixels per line and 480 lines per frame, decimation was done, that is, one pixel is captured and the two followings pixels are ignored. Then, one line is captured and the three following are ignored. The Video Buffer IN can store two lines. While one line is stored, the previous one is read from the Buffer IN by the Object Extraction Unit. The Object Extraction Unit performs all the steps for object extraction, that is, background modelling, threshold selection, and subtraction operation.

The Reference Image Unit calculates the statistical parameters by capturing the pixels from the Video Buffer IN, processing and accumulating them in the external SRAM. This operation is made for 64 frames. In order to set a threshold for pixel classification, the Threshold Selection Unit constructs a histogram of the normalized chromaticity distortion by using an internal memory of 2048 x 28 bits in the FPGA. Each position of the internal memory is associated to a value of the chromaticity distortion, which ranges from 0 to 2047 in our technique. Therefore, we calculate the chromaticity distortion, and increment the number of outcomes, registered in the internal memory position associated to that chromaticity distortion value. This operation is made for 64 frames. So, our sample space has 1.8432 x 10^6 outcomes. We set the detection rate equal to 0.99%. This means a total of 1843 outcomes of our sample space. To set a threshold, we compute the number of outcomes registered in each position of the internal memory, starting from the last position (2047), until get a sum equal to 1843. Then, the memory position where we get an amount of 1843 outcomes is the value of the threshold to be used for a pixel classification. The Pixel Classification Unit calculates the chromaticity distortion for each pixel of the current image, and compares it with the threshold selected previously. If a pixel in the current image has a chromaticity distortion greater than the threshold, it is classified as belonging to the foreground. The SRAM Interface generates the signals to read the pixels from external SRAM or to write on it. The external SRAM block has two 64K x 32 bits SRAM, UT6164C32, from UTRON. In this block the reference image, as well as the segmented image, are stored. The I2C Interface performs the configuration of the Video Decoder during the initialization. The Video Encoder Interface provides the segmented image from the external SRAM to the RGB Encoder. The Interface stores two even lines and two odd lines in the Video Buffer OUT. While one even line and one odd line are store in the Buffer, the other two lines are sent to the RGB Encoder. Just as the input image processing decimation is done, it is necessary to repeat the same pixel twice and the same line three times so that we get 720 pixels per line and 480 lines per frame again at the output. The RGB Encoder converts the digital RGB signal from the FPGA to analog RGB. For this application the THS8134B from Texas Instruments is used. Finally, the

PAL/NTSC Encoder converts the analog RGB to composed video. The TDA 8501 from Philips Semiconductors is used for this.

In Table 1 the clock frequency is calculated as a function of the horizontal and vertical resolution, and considers 35 operation cycles for each pixel. The processing time is the time necessary to process all the pixels of one line which must be done before the acquisition of the next line. The processing time and the clock frequency were computed using the equations 1 and 2, respectively.

TABLE 1: CLOCK FREQUENCY

Vertical Resolution (Lines per Frame)	Horizontal Resolution (Pixel per Line)	Processing Time (ms)	Clock Frequency (MHz)
480	360	480	199,7
240	360	240	99,36
160	240	160	44,16
120	240	120	33,12

$$PT = \frac{TVR \times HLT}{VR} \qquad (1)$$

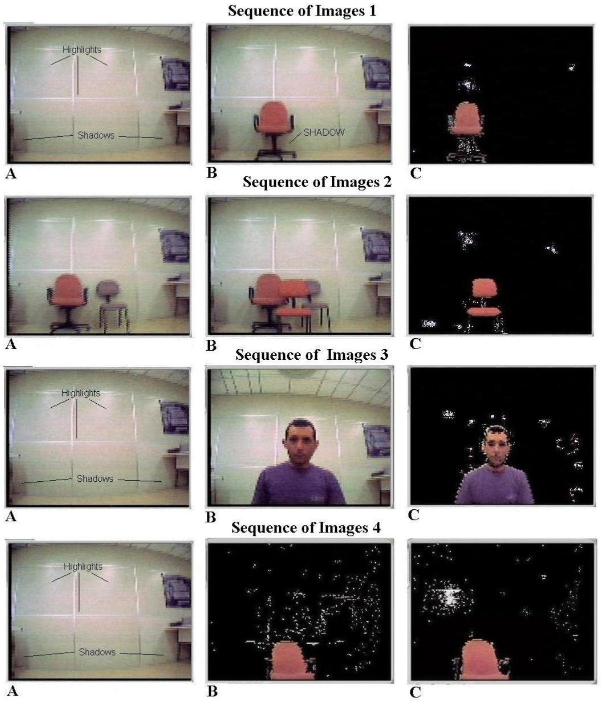
$$CF = \frac{NC \times HR}{PT} \qquad (2)$$

Where PT is the processing time; TVR is the total vertical resolution; HLT is the horizontal line duration; VR is the vertical resolution; NC is the number of operation cycles; CF is the clock frequency; HR is the horizontal resolution. From the Video Decoder specifications the total vertical resolution is 480 lines per frame. As we used PAL-M video input, the horizontal time duration is 63.4 μs. According to what is shown in Table 1 the higher the video resolution, the higher is the clock frequency. In our technique we used the clock frequency equal to 40 MHz and the resolution of 240 pixels per lines and 120 lines per frame.

3. RESULTS

This section presents the performance of the hardware set-up in a actual environment with some phenomena such as shadows, highlights, and similar colors between the foreground objects. The sequence of images shown in Figure 4 was used to evaluate it and the image resolution is 240 x 120 images. The sequences of images are organized as follows. Sequences 1 to 3: a) the original background image, b) original background image with the object to be extracted, c) resultant extraction image with the object. Sequence 4: a) the original background image, b) resultant extraction image after reducing the environmental illumination, and c) resultant extraction image after increasing the environmental illumination. The original background image contains shadows on the wall, as well as on the floor, which are generated by two tables located near the edges of the background image. There are also saturated white points due to the environmental illumination. In all the resultant images, there are white points classified as foreground pixels due to errors caused by the threshold selection, and due to noise in the signal acquired by the video decoder. The detection rate was set to 0.99. This means that for an image of 240 x 120 at least 288 pixels from the background can be classified as belonging to the foreground. In the Sequence of Images 1 the object to be extracted also generates shadows on the background image. Despite the low resolution of the system, and the imperfections in dark pixels of the object (already foreseen in the Horprasert's model) the chair was extracted from the background. The white points in the resultant image are pixels of the background classified as foreground.

The Sequence of images 2 evaluates the capacity of the system to extract an object from a heterogeneous background. There are two chairs with different colors in the background. Moreover, the object used for extraction has a color equal to one of the objects in the background. In the resulting segmentation image, it can be observed that the pixels of the object in front of the object of the background were classified as background pixels because of the color similarity between them. The pixel classification is based on the chromaticity difference between the expected values in the background image and the current values in the foreground. Therefore, pixels with similar colors are always misclassified. In the Sequence of Images 3, the white points in the resultant image are pixels of the background classified as foreground. In the boundary around the person there are many whites points because in these regions the camera can capture a pixel as belonging to the background, and in


Sequence of Images 1

Sequence of Images 2

Sequence of Images 3

Sequence of Images 4

Figure 4: Sequences of images to evaluate the object extraction system: a) original background image; b) original background image with the object to be extracted; c) resultant extracted image with the object.

another frame capture the same pixel as belonging to the foreground. Therefore, it generates a noise that makes the pixel segmentation more difficult. In the Sequence of Images 4, we change the illumination of the environment. The white points are pixels of the background classified as foreground. Most of the pixels are located in the regions of highlights in the background. Changing the illumination, the median and the standard deviation of the pixels change so much that it causes a misclassification. Therefore, an updating of the background model would be necessary to deal with those phenomena. Horpraset et al. [2] implemented their algorithm in a Pentium II – 400 MHz using floating points operations and it processed 320 x 240 images at 27.6 ms per frame. In our technique we developed specific hardware and used fixed points operation to process 240 x 120 images at a rate equal to 30 frames per second. Our main goal in this work was to propose a hardware architecture for image segmentation and demonstrate the performance of object segmentation in real time, despite the low image resolution. As shown in the Table 1, a clock frequency of 99 MHz is enough to process 360 x 240 images in the hardware at a rate of 30 frames per second.

4. FPGA SYNTHESIS RESULTS

The technique presented in this paper was distributed in two programmable logic device with 8320 logical elements each one. In device one, 91% of the available logical elements were used. In device two, 88% of the available logical elements were used. As the number of used logical elements increases, the layout becomes more complex, and therefore the internal delays increase. For this reason, the clock frequency used in the hardware was only 40 MHz. The image resolution obtained was 240 pixels per line and 120 lines per frame. The Video Decoder provides a resolution of 720 x 480. So, a sub-sampling of the digitalized input video was made before the image processing. The clock frequency for the Video Encoder was 4.5 MHz. To compute the statistical parameters during the background modelling step as well as in the segmentation step, we implemented three 24 bits adders, four 8 bits subtracters, three 10 bits x 8 bits multipliers, three 24 bits x 16 bits multipliers, three 8 bits x 8 bits multipliers, three 27 bits x 12 dividers, one 50 bits x 42 bits divider, and one 28 bits square root operator.

5. CONCLUSIONS

In this paper we proposed an architecture in FPGA for object segmentation and evaluate its performance by implementing Horprasert's algorithm. The system was subjected to phenomenon normally found in real environments, such as shadows and illumination variations. Limitations of the extraction method in the case of dark objects were observed, as already anticipated by Horprasert, and also effects due to noise in the output signal of the video decoder were identified. The system processed images in real time, that is, with a rate of 30 pictures for second. Having implemented and validated the extraction in the hardware, it was possible to show the concept that the system is capable of extracting an object from of a heterogeneous and static background in real time. Improvements are necessary to increase the image resolution, which is part of our ongoing research.

REFERENCES

[1] Alexandropoulos, Theodoros; Loumos, Vassili; Kayafas, Eleftherios. “A block-based clustering technique for real-time object detection on a static background”, 2nd International IEEE Conferenceon Intelligent Systems, Varna, Bulgaria, June 22-24, 2004.

[2] Horprasert, T.; Harwood, D.; Davis, L.S. “ A Statistical Approach for Real Time Robust Background Subtraction”. IN IEEE ICCV’99 FRAME-RATE WORKSHOP, 1999.

[3] Kim, Kyungnam; Hoprasert, Thanarat; Harwood, David; Davis, Larry. “A Background Modeling and Subtraction by Codebook Construction”. IEEE International Conference on Image Processing (ICP) 2004.

[4] Ming, Ying; Jiang, Jingjue; Ming, Jun. “Background Modeling and Subtraction Using a Local-Linear-Dependence-Based Cauchy Statistical Model”. Proceedings of the Seventh International Conference on Digital Image Computing: Techniques and Applications, DICTA 2003: 469-478.

[5] Tsai, Yu-Pao; Lai, Chih-Chuan; Hung, Yi-Ping; Shih, Zen-Chung. “A Bayesian Approach to Video Object Segmentation via Merging 3-D Watershed Volumes”. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 15, No. 1, January 2005.

[6] Mezaris, Vasileios; Kompatsiaris, Ioannis; Strintzis, Michael G. “Video Object Segmentation Using Bayes-Based Temporal Tracking and Trajectory Based Region Merging”. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 14, No. 6, June 2004.

[7] Zivkovic, Zoran. “Improved Adaptive Gaussian Mixture Model for Background Subtraction”. In Proceedings ICPR, 2004.

[8] Cucchiara, R.; Onfiani P.; Prati, A.; Scarabottolo. “Segmentation of Moving Objects at Frame: A Dedicated Hardware Solution”. Image Processing and its Applications, Conference Publication N.o 465, IEE, 1999.

[9] Kim, Jinsang; Chen, Tom. “A VLSI Architecture for Video-Object Segmentation”. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 13, No. 1, January 2003.