Implementing Computer Vision Algorithms in Hardware: an FPGA/VHDL-based Vision System for a Mobile Robot.

Reinaldo A. C. Bianchi[1,2] and Anna Helena Reali Costa[1]

[1] LTI - Laboratório de Técnicas Inteligentes, Departamento de Engenharia da Computação,
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, 158.
05508-900 São Paulo – SP – Brazil.
{rbianchi, anna}@pcs.usp.br

[2] Faculdade de Engenharia Industrial, Departamento de Engenharia Elétrica,
Av. Humberto de A. C. Branco, 3972 - 09850-901
São Bernardo do Campo, SP, Brazil
rbianchi@cci.fei.br

Abstract. A time critical process in a real-time mobile robot application such as RoboCup, is the determination of the robot position in the game field. Aiming at low-cost and efficiency, this paper proposes the use of field-programmable gate array device (FPGA) in the vision system of a robotic team. We describe the translation of well-known computer vision algorithms to VHDL and detail the design of a working prototype that includes image acquisition and processing. The CV algorithms used in the system includes thresholding, edge detection and chain-code segmentation. Finally, we present results showing that an FPGA device provides hardware speed to user applications, delivering real-time speeds for image segmentation at an affordable cost. An efficiency comparison is made among the hardware-implemented and a software-implemented (C language) system using the same algorithms.

1 Introduction

One of the time critical processes of real-time mobile robot applications, such as the RoboCup [1] or any other dynamic and interactive domain is the determination of the robot position. Several approaches have been traditionally used to solve this problem, including custom hardware accelerators and software systems.

On one hand, custom hardware accelerators for image processing are usually high priced, closed systems that implements tracking or threshold algorithms, which can deliver real-time performance but do not allow hardware reconfiguration to adapt to new situations. On the other hand, software systems using low-cost image acquisition boards present lower performance than hardware approaches and hinder robot miniaturization.

Aiming at a low-cost, small-size, hardware-based vision system we propose the use of a Field Programmable Gate Array (FPGA) device for the image processing in the F-180 and the F-2000 leagues.

This paper presents the translation of well-known Computer Vision algorithms to VHDL, a programming language used to define the function of the FPGA circuit and the design of a working prototype that includes image acquisition and processing. The CV algorithms used to define the position of the objects in an image thresholding, Edge Detection and Chain-Code Segmentation.

These algorithms were tested on an ALTERA Laboratory Package and on individual components. Testing results shows that FPGA device provides hardware speed to user applications, delivering real-time speeds for image segmentation at an affordable cost (lower than a custom PC frame grabber).

The reminder of this paper is organized as follows. Section 2 introduces the FPGA and the VHDL language used to write programs that define the FPGA circuit. Section 3 describes the complete vision system, including the image acquisition process and the CV algorithms used. Section 4 presents the translation of the algorithms to VHDL. Section 5 presents the experimental setup and the results obtained. Finally, Section 6 summarizes some important points learned from this research and outlines future work.

2 FPGA and VHDL

Field Programmable Gate Arrays – FPGA [2] are integrated circuit that can have their hardware designed by the user. An FPGA contains a large number of identical logic cells that can be viewed as pre-defined components, which combines a few inputs to one or two outputs according to a Boolean logic function specified by a user defined program. In its turn, individual cells are interconnected by a matrix of wires and programmable switches. The advantage of using a FPGA is that it gives hardware speed to user applications.

A user's design is implemented by specifying the logic function for each cell and selectively closing the switches in the interconnect matrix. This is done by means of a user program that defines the function of the circuit, usually written in VHDL.

VHDL [3] stands for VHSIC Hardware Description Language, where VHSIC means Very High Speed Integrated Circuits. VHDL is a language used to model a digital system, from a simple gate to a complete system.

3 Description of the System

Several approaches have been used to segment the images in the RoboCup domain. The following modules (see figure 1) compose the vision system implemented: image acquisition, thresholding, edge extraction, and chain-code segmentation.

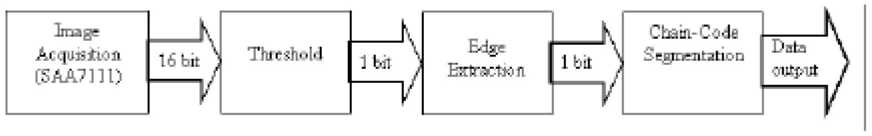

Fig. 1. Block Description of the vision system.

3.1 Image Acquisition

The image acquisition in the developed system is made through the Phillips' SAA7111 Video Input Processor. It is a low cost CMOS that receives analogical video signal and transforms it in a digital output.

Its architecture combines a two-channel analog preprocessing circuit (with an anti-aliasing filter, an automatic clamp and gain control), a digital multi-standard decoder and a brightness/contrast/saturation control circuit. The decoder is based on the principle of line-locked clock decoding and is able to decode TV signals (PAL BGHI, PAL M, PAL N, NTSC M and NTSC N) into CCIR-601 compatible color component values. In the vision system, the analog input is an NTSC N CVBS signal and the digital output is an RGB 16-bit signal.

3.2 Image Processing Algorithms

The implementation of edge following algorithms based on Freeman chain coding [4] was chosen because they provide area, perimeter, center of area, minimum enclosing rectangle, orientation, shape factor and other valuable classification information. In this section, algorithms to perform binary image edge following algorithms are presented.

3.2.1 Image Thresholding

The digital RGB 16 bit signal that composes the color image is converted to a 1 bit signal by a simple color detection criteria: if the R, G and B values are in the desired range, the output is true. Else, the output is false. This criteria is used by several researchers [5] and generates a one bit image that will be used by the edge detection algorithm.

3.2.2 Edge Detection

This algorithm apply an edge extracting operator in the image, replacing all the white pixels in the binary image by black ones, unless they are in a black/white border. The operator verifies four elements (2x2 mask) to determine if a point is in a border. Table 1 presents six basic configurations of the sixteen possible combinations of four elements of a binary image.

3.2.3 Edge following

This algorithm, through a procedure that follows the borders in the image, creates a current of connected vectors (called chain) that envelops the objects in the image. The perimeter can be mesured as the sum of the constituent vectors, while the area is mesuring by summing the areas between each vector and a reference line (a procedure similar to area integration). Moment of area and centroids can be found by similar calculations. Finally, a shape factor, defined as (area)/(perimeter) are computed. Figure 2 presents the result of the three steps aplied to the image of a ball.

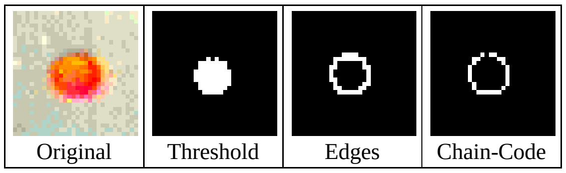

Fig. 2. The result of the image processing steps.

3.3 Data output.

The information about the objects in the image is stored on elements of an array with 32 bits elements, where 18 bits represents the position where area center of the object is located (9 bits for the line and 9 for the column) and 14 bits register the size of the minimum enveloping chain. This information can be sent to another FPGA module such as the strategy module or to a RS-232 transmission module.

4 Translation to VHDL

While implementing the project the three modes to write a VHDL specification described in section 2 were combined in the so-called mixed-mode design, where the designer can implement part of the system just like a digital circuit and another part just like an algorithm. As the chain-code algorithms have a considerable size, we present here the simplification of the thresholding device.

The first part of the code defines the input and output signals, where there are 5 bits for the red and blue signals and 6 bits for the green signal, and only one bit for the output signal. The second part defines the function: a threshold that will accept only colors with the red, green or blue with half the maximum intensity. If any of the most significant bits is on, the output signal will be off.

Example of a VHDL program that implements a simplified thresholding device using the dataflow method.

```
-- This part defines the input and output ports

Entity Threshold is
```

```
   PORT ( R, B: IN BIT_VECTOR(0 TO 4);
          G :   IN BIT_VECTOR(0 TO 5);
          I :   OUT BIT
          );
End Threshold;

-- This part defines the device function
Architecture Threshold of threshold Is
BEGIN
   I <= not (R(4) or B(4) or G(5));

End Threshold;
```

5 Experimental Results

The algorithms were tested using the simulation tool of the ALTERA MAX+PLUSII software and on an Educational Board (figure 3). The simulation results of a 100 MHz FPGA device implementing the vision algorithms, for a 320 x 240 pixels color image, are presented in the table below.

Table 1. Time needed by each module to process an image with 320 x 240 pixels.

Module	Time
Thresholding	770µs
Edge Extraction	2,41µs
Chain-Code Segmentation	770µs
Total time	1,54ms

The table shows that the edge extraction is the fastest module. This happens because its implementation processes a line at each cycle time. As the other modules takes one cycle time to process each pixel their time are the same. Also, these results were obtained assuming that modules are working sequentially: the edge extraction starts when the image is binarized and the chain-code begins when the edge extraction finishes.

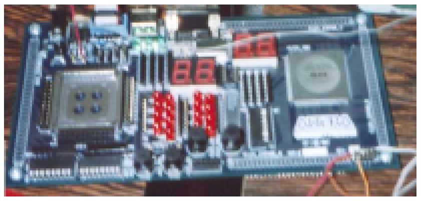

Fig. 3. The ALTERA Education Board.

Finally, this table shows that the system can process 649 images (with 320 x 240 pixels) per second. Comparing with a straight forward implementation of the same algorithms in C Language, working on a Pentium 200 computer with Linux OS, the software took 33ms to process each image (21,5 times the FPGA time).

As for the quality of the results, the resulting images from the FPGA were similar to the ones obtained using the software implementation.

6 Conclusion

This paper presented the translation of well-known Computer Vision algorithms to VHDL and a working prototype that includes image acquisition and processing.

An efficiency comparison among the hardware-implemented and a software-implemented (C language) system using the same algorithms showed that the first system achieved a superior performance with the same quality. Associated with the low cost of FPGA components (lower than a Video for Windows PC frame grabber), the resulting system is a small-size device suited for the image processing in the F-180 and the F-2000 leagues.

Future works include the implementation of other Computer Vision algorithms such as Blob Coloring in VHDL and final tests of the system on board a F180 Robot Prototypes and Activemedia Pioneer 2 robots.

7 Acknowledgments

This project is supported by Fundação de Ciências Aplicadas – FCA-FEI under grant number OS-5886 and by NSF/CNPq-ProTeM CC, project MAPPEL, grant number 68003399-8. The authors would like to thank the undergraduate students who took part in this project: Luis Maia Jr.., Paulo V. de Souza, Victor Antônio, and Marcel Gomes.

References

1. Kitano, H. et al.: RoboCup: A challenge Problem for AI. AI Magazine, Vol. 18, No. 1, Spring (1997) 73-85
2. Brown, S.; Rose, J.: Architecture of FPGAs and CPLDs: A Tutorial. IEEE Design and Test of Computers, Vol. 13, No. 2 (1996) 42-57
3. Bhasker, J.: A VHDL Primer. Prentice Hall, Englewood Cliffs (1995)
4. Kitchin, P. W., Pugh, A.: Processing of Binary Images. In: Pugh, A. (ed.): Robot Vision. Springer-Verlag, Berlin Heidelberg New York (1983) 21-42
5. Brusey, J.; Padgham, L.: Techniques for Obtaining Robust, Real-Time, Colour-Based Vision for Robotics. In: Veloso, M., Pagello, E, Kitano, H. (eds.): RoboCup-99: Robot Soccer World Cup III. Lecture Notes in Artificial Intelligence, Vol. 1856. Springer-Verlag, Berlin Heidelberg New York (2000) 243-253