Algorithm and Hardware Development for Pupil Measurements and Analysis in Digital Images for Medical and Other Applications

Ricardo U. Pedroni, Volnei A. Pedroni

CPGEI – Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial
UTFPR – Universidade Tecnológica Federal do Paraná
`pedroni@utfpr.edu.br`

Abstract— **Computer vision is increasingly becoming a common resource in tasks that before were only accomplishable through human-based evaluations. In the healthcare area, digital image processing has helped doctors in diagnosis procedures and patient treatment. The use of pupillometry, the measurement of pupil size under different incident light intensities, is a non-intrusive method that serves, aside other possible uses, to evaluate diabetic autonomic neuropathy (DAN) in diabetic patients. For that, a digital image processing algorithm is being developed, embedded on an FPGA-based hardware with the objective of creating a portable device capable of computing pupil size in real-time on a low to medium cost system, comfortable and easy to use for doctors and patients alike**.

Keywords— **Pupillometry, image processing, biomedicine, FPGA.**

I. INTRODUCTION

Computational vision is a useful resource in various segments of human activities such as industry, commerce, and healthcare. In healthcare, digital image processing technology is used mainly as a source of auxiliary information to the health-related professionals (i.e. doctors), where information provided by a machine is somehow reported and, based on this information and the doctor’s experience, a patient’s diagnosis is elaborated. More recently, the development of personal-use equipment has been considered, such that a doctor’s analysis is no longer needed, creating results that can be interpreted directly by the patient.

Such equipment tends to cheapen clinical costs as a whole in a mid to long term time span, since no specialist is required to evaluate the results. For such, though, the exam results must be of the most utter precision with a very low error rate since no healthcare specialist will aid the patient in understanding the output result.

Amid the various possible medical uses for this type of digital image processing, one is *optical biometry* (OB). OB works as follows: a digital image (or a sequence of images) of the patient’s eye is acquired and an algorithm obtains the relevant parameters from this image (e.g. pupil size, iris size, size variation with time, etc.). These algorithms are built upon one or a combination of image processing mathematical and analysis tools, such as image edge detection [1] and neighborhood pixel average.

An example of a medical situation that pupillometry has an important use is in the analysis of diabetic autonomic neuropathy (DAN) for patients that suffer from diabetes. It is known that the pupil dilates and retracts based on different occurrences of light intensities. This change of size is produced by the sympathetic and parasympathetic autonomic nervous system. It has been proven [2] that patients whom suffer of DAN portray different pupil size variations to certain light stimulus when compared to people who do not have this disease. Given this information, a precise pupillometer may be a way to diagnose the presence of this disease.

Another situation that pupillometry can reveal itself useful is in the detection of drug-influenced persons. The consumption of drugs modifies pupil reaction to light. Certain drugs cause reduction of pupil size [3], like alcohol consumption. Other drugs, such as LSD and cocaine may be responsible for abnormal pupil dilation. Law-enforcing officials may use pupil measuring apparatuses to confirm that an individual is under influence of a certain substance through a non-invasive matter.

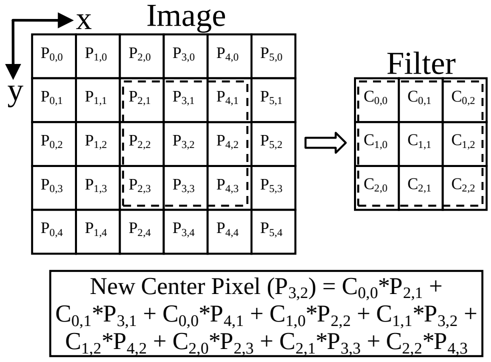
Due to these possible uses of pupillometry, a project has been started with aim on completing a robust pupillometer that can fulfill the uses described above and possibly others. In the present paper, the objective is to explain the algorithm being developed and the steps taken to embed this algorithm on an FPGA-based system, to create reliable pupillometry equipment.

In section 2, a theoretical background is presented, detailing the mathematical tools and image processing resources used within the algorithm. In section 3, the algorithm itself is explained, showing its overall form and how it works. In section 4, the chosen hardware is detailed, demonstrating how the algorithm is being transposed to a Hardware Description Language (HDL). Hardware block diagram figures will further aid the explanation. In section 5, a brief demonstration of results provided through MatLab® will be presented. Finally, in section 6, conclusions regarding the current state of the project will be presented, along with remarks on related future work.

II. THEORETICAL BACKGROUND

Edge detectors are algorithms that attempt to extract from a digital image the borders of objects present in this image, be them colored or grayscale. The algorithm created by Canny [3] serves as a robust edge detector that works fairly well even for noisy data sets, as is the case with simpler CCD cameras.

The first stage of the Canny edge detector is to blur the input image, with the main purpose of reducing noise. This can be accomplished by a low pass filter, such as a Gaussian filter. The filter is applied as a sliding window, normally of uneven size (e.g. 3x3, 5x5). A convolution occurs between the filter and the image, that is, pixel intensity values are multiplied by the value of each respective coefficient of the filter and all values are summed, forming the value of the new output pixel.

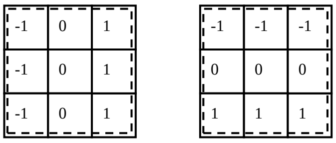


$$\begin{array}{|c|c|c|c|c|c|}\hline P_{0,0} & P_{1,0} & P_{2,0} & P_{3,0} & P_{4,0} & P_{5,0} \\ \hline P_{0,1} & P_{1,1} & P_{2,1} & P_{3,1} & P_{4,1} & P_{5,1} \\ \hline P_{0,2} & P_{1,2} & P_{2,2} & P_{3,2} & P_{4,2} & P_{5,2} \\ \hline P_{0,3} & P_{1,3} & P_{2,3} & P_{3,3} & P_{4,3} & P_{5,3} \\ \hline P_{0,4} & P_{1,4} & P_{2,4} & P_{3,4} & P_{4,4} & P_{5,4} \\ \hline \end{array} \Rightarrow \begin{array}{|c|c|c|}\hline C_{0,0} & C_{0,1} & C_{0,2} \\ \hline C_{1,0} & C_{1,1} & C_{1,2} \\ \hline C_{2,0} & C_{2,1} & C_{2,2} \\ \hline \end{array}$$

New Center Pixel ($P_{3,2}$) = $C_{0,0}*P_{2,1} + C_{0,1}*P_{3,1} + C_{0,0}*P_{4,1} + C_{1,0}*P_{2,2} + C_{1,1}*P_{3,2} + C_{1,2}*P_{4,2} + C_{2,0}*P_{2,3} + C_{2,1}*P_{3,3} + C_{2,2}*P_{4,3}$

Fig. 1 Example of filtering through convolution

The second stage of the algorithm consists on finding the gradient of the filtered image. Recall that the gradient of an image is the directional change of intensity. A higher change in absolute intensity produces a higher gradient value. Such a filter as a Prewitt filter can accomplish this task. The purpose of finding the gradient is that edges commonly represent high changes in intensity value, so the gradient will be a fair initial estimate of detected edges.



$$\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Fig. 2 Prewitt x and y gradient filters

Right after the gradient calculation, the non-maximum suppression stage is applied. This stage calculates if the gradient magnitude assumes a local maximum in the gradient direction, based on the neighboring intensity values following this direction. If the magnitude is larger then these neighboring values, then it is a local maximum and is marked as so. Values that fulfill the opposite condition (are smaller) are marked as not being edges and have their intensity zeroed.

Finally, the resulting intensity map is thresholded. A low (T_L) and a high threshold (T_H) values are chosen. Every intensity that was not zeroed in the previous stage is compared with these thresholds. If the intensity value is higher than T_H, this value is marked as a true edge. If the intensity value falls bellow T_L, it is zeroed. Values found in between both thresholds are compared with neighboring values. If any neighbor is a true edge, then this value is also marked as a true edge.

The Canny edge detector will be one of the stages in the project's final algorithm.

III. ALGORITHM

This project aims at accomplishing a low to mid cost device that can accurately output pupil size readings. To achieve this, higher cost hardware becomes very restrictive. Since modern technology, in a general matter, offers better performance components at higher prices, a middle term performance component is used.

Due to this fact, extremely fast and/or heavy resource usage algorithms can’t be used. A workaround to this is to choose an algorithm that achieves the demanded objective (i.e. accurate pupil measurement), that is simultaneously a computationally light process and can process all this information in a reasonable time. After thorough studies of algorithms and manners this could be accomplished, the final algorithm was chosen and is now explained.

Before applying the algorithm, certain parameters are chosen. These values vary from Gaussian filter standard deviance to probable maximum pupil size, amidst others. Their main purpose is to give the algorithm maximum flexibility and configuration. This way results can be tweaked and optimized.

The first stage consists of obtaining the digital image (in our case, through a CCD camera). This image is of 640x480 pixels. Once this image is acquired, a Canny edge detector is applied, one such as the one explained in the previous section. The output of this stage is a binary image, consisting of ones where calculated edges are located.

Stage two of the algorithm processes this binary image, finding all borders present in the image. Borders are fragments of the image that are connected. If a pixel is marked as one and any of its 8-neighbors is also one, these two pixels are considered connected. Following this logic for all pixels, all borders in the image can be found.

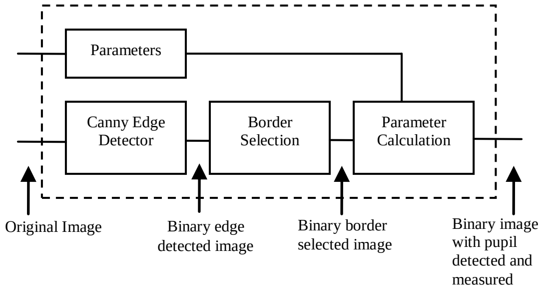


Fig. 3 Block Diagram of Algorithm

With the borders in hand, comes stage three. This will be the main body of the algorithm. One by one, every border calculated by the previous stage is tested. Various parameters are calculated, such as center of mass, distance of each border point to the center of mass, the mean and variance of these points. Based on these values and the parameters values chosen initially, if the border attends all parameters, it is marked as the pupil and the algorithm is done.

The calculation is done through a computationally light process, which compensates heavier calculations using configurable parameters that are optimized to find desired results.

IV. HARDWARE IMPLEMENTATION

As explained earlier, certain limitations existed when choosing which hardware to use. Since reasonable speed is a requirement and cost can't be greatly increased, the final main hardware choice arose as an FPGA. Due to its configurability, hardware level speed, high input/output pin count, overall easy driver interfacing with other components and such other qualities, an FPGA proved itself to be the right initial choice. The chosen model is an Altera Cyclone II EP2C35F672C, found on an DE2 development kit.

Further components include a 5MPixel CCD camera and 4.3" LCD, both from Terasic and SDRAM and SRAM memories found on the DE2 board. The CCD camera obtains the image, configured to be at a 640x480 pixel resolution. The SDRAM and SRAM serve as a image buffer for the LCD, which displays the processed final image.

The process of transforming high level code to Hardware Description Level code (i.e. at hardware level) is not a simple one. Many image processing algorithms are based on a computer and use high-level languages such as C++ and Java. Such languages present various conveniences in terms of writing code, conveniences that are not present when writing hardware-level code.

The advantage, though, of writing at such a low level is that hardware blocks, such as flip flops and shift registers can be easily implemented. Also, since the algorithm is running on pure hardware, no software delays are present, compensating coding complexity for the advantage of speed and configurability.

The project is being written in VHDL and Verilog alike.

V. SOFTWARE TESTS

Since all the code to be written in hardware has been implemented and proven it's functionality in software, utilizing MatLab®, these results shall be presented now. The image used for the testbench is from a conventional digital camera focused on the patient's eye.

The input image (see Fig. 4) is processed and the borders detected through the Canny algorithm. With the borders detected (Fig. 5), the remaining part of the algorithm can be applied. The detected pupil (Fig. 6) is the empty area inside the red circle while the iris is all the area under that same red circle. As can be seen, the result is accurate.

Fig. 4. Original test image

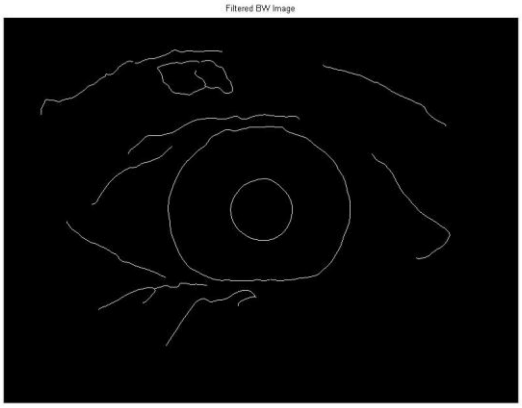


Fig. 5 Detected borders image

Fig. 6 Detected borders image

VI. CONCLUSIONS

At the current state of the project, the algorithm has been fully tested on a computer, utilizing MatLab®, and it functions as expected. Presently, the hardware implementation is at the Border Selection stage, with results equally fulfilling requested levels. The output is functioning at about 20 processed frames per second.

Further improvements will be made so a higher output frame rate can be achieved. Study of optimal parameter values will be made to achieve the best possible pupil readings.

REFERENCES

[1] J. Canny, “A computational approach to edge detection,” IEEE. Trans. Pattern Anal. Machine Intell., vol. PAMI-8, pp. 679-698,. Nov. 1986.

[2] G. L. Ferrari, “Pupilometria Dinâmica: Aplicação na Detecção e Avaliação da Neuropatia Autonômica Diabética e Estudo entre a Resposta Temporal da Pupila ao Estímulo Visual e a Glicemia”, Doctoral Thesis, UTFPR, Feb. 2008.

[3] S. H. Snyder, “Drug and Neurotransmitter Receptors in the Brain”, Science, Vol 224, Issue 4644, 22-31, 1984

[4] A. Z. Rovani, “Sistema para Captura Automática de Imagens de Íris”, MSc. Dissertation, UTFPR, Nov. 2006.