VLSI IMPLEMENTATION OF EDGE DETECTION OPERATION FOR IRIS IMAGES

Biju B[1] , Godwin N[2] , Dhanush Ben B[3], Ananthu P Nair[4] and S Absa[5]

[1]*Electronics and Communication Engineering, St. Xavier's Catholic College of Engineering, India*
E-mail: bijubyju8@gmail.com

[2] *Electronics and Communication Engineering, St. Xavier's Catholic College of Engineering, India*
E-mail: godwinsxcce@gmail.com

[3] *Electronics and Communication Engineering, St. Xavier's Catholic College of Engineering, India*
E-mail: dhanushbenjohn2002@gmail.com

[4] *Electronics and Communication Engineering, St. Xavier's Catholic College of Engineering, India*
E-mail: ananthup165@gmail.com

[5] *Electronics and Communication Engineering, St. Xavier's Catholic College of Engineering, India*
E-mail: absa@sxcce.edu.in

Abstract

In image processing, edge detection is a crucial stage, particularly in iris recognition systems and related tasks like feature extraction, segmentation, and object identification. When detecting edges, the Sobel technique is frequently used to pinpoint borders where the image's gradient is high. This project suggests using the Zynq UltraScale+ MPSoC board with MATLAB-Simulink integration to accomplish the edge detection operation for iris pictures in a VLSI. Because iris scans have distinct and trustworthy biometric properties, they are selected as input. Nevertheless, current eye detection techniques frequently encounter difficulties in different lighting scenarios. By recognizing edges in colored and other filtered photos in addition to grayscale ones, the project seeks to address these problems. MATLAB-Simulink is used to convert RGB images to grayscale, and the Vivado IDE uses Verilog code to implement the Sobel edge detection algorithm. The recommended method combines Simulink with HDL blocks, converting the Simulink model into HDL code of Verilog or VHDL with a testbench to be implemented on FPGA boards via Vitis Model Composer. Experiments show that even under difficult circumstances, this technique can precisely and successfully identify the iris edge.

Keywords — Iris recognition, Hardware description language block, Simulink, Vitis model composer.

1. INTRODUCTION

Iris recognition is a sophisticated and reliable biometric verification method that analyzes unique patterns in an individual's iris to verify their identity. The colored part of the eye is the iris, which is located between the cornea and the lens. The intricate patterns develop randomly in the womb and remain constant throughout an individual's life, giving rise to a singular and non-repeatable trait. The iris offers multiple unique features for precise identification, unlike fingerprints or other face features that are malleable or alterable. Passwords and PINs are examples of traditional identity verification methods that have long been used to safeguard digital assets and data.However, these methods are vulnerable to various security flaws like phishing, social engineering, and brute force attacks. Moreover, they rely on users' ability to remember complex passwords or PINs, which may lead to issues with usability and increased risks of unwanted access. The efficiency and user-friendliness of fingerprint and facial recognition technologies have contributed to their growing popularity in recent years. They do, however, have certain disadvantages. Fingerprints can be affected by aging, manual labor, injuries, and other factors, which can lead to erroneous rejection or acceptance rates. Even though they are widely used, facial recognition systems are susceptible to spoofing tactics that use masks or images. However, iris recognition has many advantages that make it the best option for safe authentication. The iris has several different features, including crypts, furrows, and trabeculae, which are unique to each individual and do not alter over time. This stability ensures consistent outcomes despite in shifting lighting or environment conditions. Furthermore, because iris recognition technology requires a living, unmodified iris in order to create a successful match, it is very difficult to fool or manipulate. Using VLSI-based iris identification systems has advantages in terms of increased energy efficiency, security, and performance. Through the delegation of computationally intensive operations to dedicated hardware accelerators, real-time processing rates and low-latency authentication answers can be achieved.

Furthermore, VLSI chips can be integrated into a wide range of devices, including smart cards, smartphones, and Internet of Things devices, because to their tiny size and energy efficiency. The VLSI realization of secure iris localization, a crucial step in iris identification, is the focus of this work. We propose an approach to iris localization that leverages Vivado's Vitis Model Composer to translate Simulink HDL blocks into Verilog code for FPGA execution. This technique ensures the real-time analysis of iris location and facilitates efficient hardware execution. By addressing the problems and limitations of existing localization algorithms, this VLSI-based approach offers a promising solution for safe and effective iris localization in biometric systems.

2. LITERATURE SURVEY

This section includes reviews of earlier studies on FPGA-based Sobel edge detection algorithm implementations. The reviewed papers cover a wide range of topics, including creative architectures, hardware implementations, performance assessments, and efficient filter structures. These studies support edge detection in FPGA-based systems with their insights on resource usage, real-time applications, and optimization techniques.

C. Pradabpet et.al., (2009) introduced an efficient filter structure for multiplier-less Sobel edge detection [1]. Presented at the IEEE International Conference on Signal Processing Systems (ICSPS). Their work focuses on optimizing the Sobel edge detection algorithm by designing a filter structure that eliminates the need for multipliers, aiming to reduce hardware complexity and resource utilization in FPGA-based implementations of edge detection. This innovation offers a significant advancement in FPGA-based edge detection by providing a multiplier-less architecture tailored specifically for Sobel edge detection tasks, thus contributing to more efficient and resource-conscious solutions in FPGA-based systems.

C. Perra et.al., (2005) proposed a method for image blockiness evaluation using the Sobel operator, in the EURASIP Journal on Applied Signal Processing [2]. The paper introduces a novel application of the Sobel operator for assessing the blockiness artifacts in digital images. By analyzing the gradient variations across image blocks, the proposed method offers a quantitative measure of blockiness, which is crucial for evaluating image compression algorithms and video quality. This work contributes to the field by providing a practical tool for assessing image quality and identifying compression artifacts in digital media.

D. Alghurair, S. S. AI-Rawi et.al., (2013) proposed a Sobel operator using Field Programmable Gate Array (FPGA) [3]. Published in the International Journal of Computer Applications. This work focuses on implementing the Sobel operator on FPGA for edge detection in digital images. By leveraging FPGA technology, the authors aim to achieve high-performance edge detection suitable for real-time applications. This research contributes to advancing FPGA-based image processing algorithms and their practical implementations.

Fernando Martinez Vallina, Christian Kohn, and Pallav Joshi et.al., (2012) proposed a Zynq All Programmable SoC Sobel filter implementation using the Vivado HLS tool [4]. In the Proceedings of the International Conference on Field-Programmable Technology (FPT). The paper discusses the implementation of a Sobel filter on a Zynq All Programmable SoC platform using Vivado HLS, aiming to achieve efficient edge detection in real-time applications. This work contributes to the field by demonstrating the feasibility of using Zynq SoC devices for implementing complex image processing algorithms such as the Sobel filter.

I. Yasri, N. H. Hamid, and V. V. Yap et.al., (2008) proposed a performance analysis of an FPGA-based Sobel edge detection operator [5]. Published in the International Journal of Computer Science and Network Security. The paper evaluates the efficiency of FPGA-based implementations of the Sobel edge detection algorithm. By analyzing factors such as speed, resource utilization, and power consumption, the authors provide insights into the suitability of FPGA technology for edge detection applications. This research contributes to understanding the performance characteristics of FPGA-based edge detection systems and informs their optimization for various applications.

Jamie Schiel, Andrew Bainbridge-Smith et.al., (2015) proposed efficient edge detection on low-cost FPGAs [6]. In the Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT). This work discusses methods for optimizing edge detection algorithms for implementation on low-cost FPGAs, aiming to achieve real-time performance while minimizing resource utilization. By leveraging inexpensive FPGA platforms, the authors aim to democratize access to high-performance edge detection capabilities. This research contributes to making FPGA-based edge detection more accessible for a wide range of applications.

A. Pujare, P. Sawant, H. Sharma, and K. Pichhode et.al., (2020) proposed a hardware implementation of the Sobel edge detection algorithm [7]. Presented in ITM Web of Conferences. This work focuses on the hardware implementation of the Sobel edge detection algorithm, aiming to achieve efficient and real-time edge detection capabilities. By leveraging hardware acceleration techniques, the authors aim to enhance the performance of edge detection systems for various applications.

K. Wong, A. Chekima, J. Dargham and G. Sainarayanan et.al., (2008) proposed palmprint identification using the Sobel operator [8]. Published in Pattern Recognition. The paper introduces a method for palmprint identification based on the Sobel operator, aiming to achieve accurate and reliable biometric authentication. By analyzing the unique patterns present in palmprint images, the proposed method offers a robust identification technique suitable for security applications. This research contributes to advancing biometric authentication methods based on image processing techniques.

Santanu Halder et.al., (2012) proposed a fast FPGA-based architecture for Sobel edge detection [9]. Published in the International Journal of VLSI Design & Communication Systems (VLSICS). This work presents a fast FPGA-based architecture for Sobel edge detection, aiming to achieve high-speed and efficient edge detection capabilities. By optimizing hardware resources and algorithmic implementations, the proposed architecture contributes to the development of real-time edge detection systems.

Tanvir A Abbasi, Mohd Abbasi et.al., (2007) proposed a FPGA-based architecture for Sobel edge detection operator [10]. Published in the International Journal of Computer Science and Network Security. The paper presents a proposed FPGA-based architecture tailored for implementing the Sobel edge detection operator efficiently. By exploiting FPGA's flexibility and parallelism, the proposed architecture aims to enhance edge detection performance, making it suitable for applications requiring real-time processing.

3. METHODOLOGY

Through the VLSI implementation of the edge detection operation for iris images, the research aims to improve the speed and accuracy of authentication operations in real-time iris recognition systems.

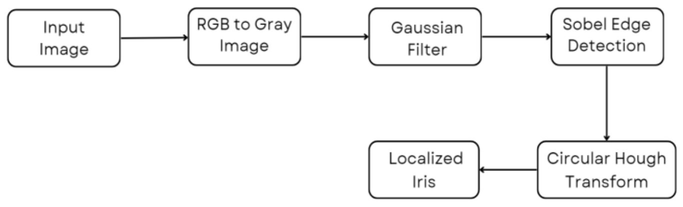


Fig.3.1 System block diagram

The block diagram of the project flow, as shown in Fig.3.1, outlines the essential procedures for improving the precision and speed of authentication procedures in real-time iris recognition systems by implementing VLSI.

3.1 IMAGE ACQUISITION FROM INPUT

Getting an input image with the eye region included is the first step. To ensure clear and accurate photos, this picture was taken with a specialized camera—such as an iris scanner—in a controlled lighting environment. However, images can be selected from a specific database. The image is first converted to a single-dimensional array, and each pixel value is recorded separately as a red (R), green (G), and blue (B) element.

3.2. PRIOR TO PROCESSING

RGB to Gray Conversion: Simulink's HDL blocks are used to transform the RGB input image to grayscale. Processing is made simpler and computational complexity is decreased in this step. Using adder and multiplier operations based on the luminance formula:

$$Gray = 0.2126 \times R + 0.7152 \times G + 0.0722 \times B.$$

3.3. IMAGE ENHANCEMENT

The technique of altering digital photographs to make the results more appropriate for display or additional image analysis is known as image enhancement. To make it simpler to recognize important aspects in an image, you can, for instance, brighten, sharpen, or remove noise.

3.4. DETECTION OF EDGES

One common technique in image processing for finding edges in an image is called Sobel edge detection. In order to highlight areas of significant intensity variation, the Sobel operator computes the gradient of the image intensity at each pixel. Two 3x3 kernels are used by the Sobel operator; Sobel_x is used to identify edges in the horizontal direction, while Sobel_y is used to detect edges in the vertical direction.

To get the approximate gradient of the image intensity in the x and y directions, respectively, these kernels are convolved with the image. The gradient's direction shows the edge's orientation, and its magnitude reflects the strength of the edge at each pixel.

The following are the kernels for Sobel_x and Sobel_y:

$$Sobel\_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$Sobel\_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

The image is convolved with these kernels individually to compute the gradient at each pixel. The gradient's magnitude is then computed as follows:

$$Gradient\ Magnitude = \sqrt{(Sobel\_x * Image)^2 + (Sobel\_y * Image)^2} \qquad (3.1)$$

* indicates the convolution procedure in this case. One can compute the gradient's direction as follows:

$$GradientDirection = \arctan2\,(Sobel\_y * Image, Sobel\_x * Image) \qquad (3.2)$$

Then, edges in the image can be identified using the gradient's amplitude and direction; larger magnitudes denote stronger edges.

3.5 IMAGE OUTPUT

Once the iris picture has been subjected to the Sobel edge detection procedure in Vivado IDE, the processed image can be accessed for additional analysis and authentication. This processed image is usually saved in a special directory for processed image storage on the target hardware platform's filesystem.

A Python script is used to gain access to the target hardware platform's file manager in order to retrieve this processed image. The script navigates to the precise directory containing the processed image by using Python's file management modules, such as shutil or os. After the script finds the processed picture file, it reads and loads the image data into memory using libraries for image processing in Python, like Pillow (PIL) or OpenCV. This stage makes it possible to transform the processed image into a format—such as numerical arrays or matrices—that is appropriate for additional analysis. After the picture data has been loaded, it can be used for a number of purposes, such as feature matching, iris pattern extraction, and machine learning-based authentication. For authentication reasons, the picture data can be processed to extract distinctive iris features or compared to a pre-registered iris template.

3.6 BENIFITS

Efficient chip modeling in VLSI-based iris recognition systems allows for easy integration with current systems and improves interoperability with different security protocols. Furthermore, VLSI chips' low power consumption guarantees great performance over an extended period of time in portable devices like cellphones and smart cards. Together, these characteristics highlight the usefulness and adaptability of VLSI-based iris recognition systems, positioning them as optimal solutions for real-world applications demanding robust and efficient biometric authentication.

4. RESULT

4.1 Synopsis

The recommended method effectively handles picture data by utilizing hardware description language (HDL) blocks in Simulink and MATLAB. This demonstrates the method's versatility and efficacy. The Zynq UltraScale+ FPGA board can more easily implement complex algorithms because to the seamless integration of software and hardware design tools, as demonstrated by the use of Vitis Model Composer to convert the Simulink model into Verilog code.

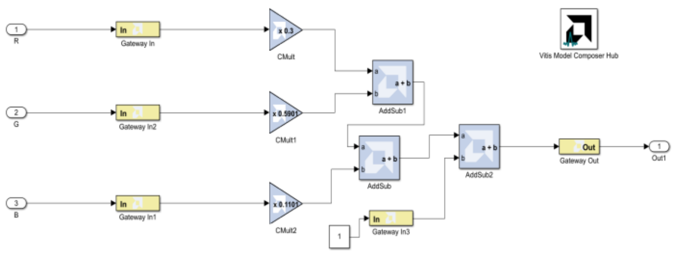
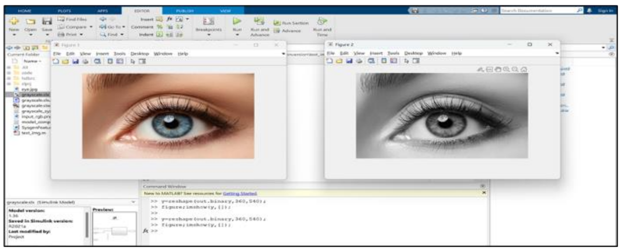


Fig.4.1.1 RGB to Grayscale Conversion using Hdl blocks

The procedure for converting an RGB image to grayscale using Simulink's HDL blocks is shown in Fig. 4.1.1. The picture shows how color pixels are converted to a grayscale intensity representation.

The project uses Verilog code to incorporate Sobel edge detection and Gaussian blur techniques, demonstrating the potential for effective image processing on FPGA devices.

These techniques are essential for enhancing the quality of images and locating significant features required for iris localization. The Gaussian blur technique effectively reduces noise and smoothes the image, while the Sobel edge detection algorithm finds edges exactly, which helps identify the limits of the iris with accuracy.



```
>> y=reshape(out.binary,360,540);
>> figure;imshow(y,[]);
>>
>> y=reshape(out.binary,360,540);
>> figure;imshow(y,[]);
>>
```

Fig. 4.1.2 Test image: RGB to Grayscale

A test image is displayed in Fig. 4.1.2 both before and after the RGB to grayscale conversion. It shows what happens when a color image is converted to grayscale. Converting the grayscale image to a binary file demonstrates the accuracy and robustnessof the used techniques.

This successful conversion confirms the efficacy of the applied image processing techniques and provides a solid foundation for the iris identification processes to follow. When these methods are used, image processing duties can be efficiently managed by FPGA-based systems, which improves iris recognition applications' dependability and performance.

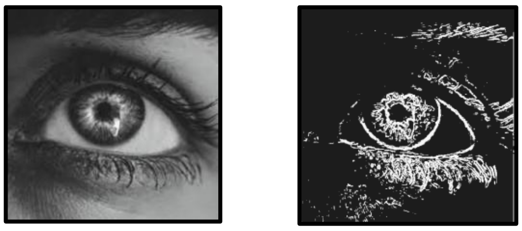


Fig.4.1.3 Edge Detection Test-1: Greyscale to Figure

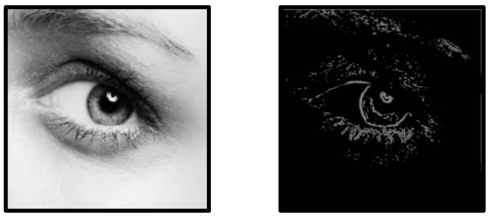


Fig.4.1.4: Edge Detection Test-2: Greyscale to Figure

The images are displayed in Figs. 4.1.3 and 4.1.4 following the use of the Sobel edge detection technique. It draws attention to the margins found in the pictures, which are essential for recognizing characteristics in iris localization.

The outcomes of the edge detection and RGB to grayscale conversion phases demonstrate how successful the recommended strategy is. The resulting grayscale images provide a clear and accurate depiction of the iris region, making them perfect for iris detection. Furthermore, the pictures that are produced by utilizing Sobel edges exhibit precise edge recognition, which is crucial for precisely delineating the boundaries of the iris. These outcomes validate the method's efficacy and offer a solid foundation for iris recognition advancements.

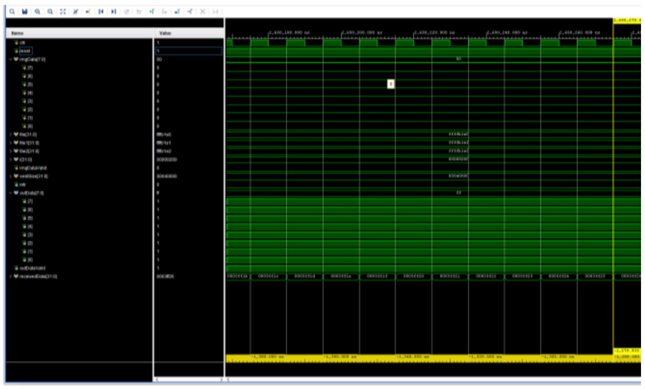


Fig. 4.1.5 Simulation waveform output obtained by using Sobel operation for edge detection

The simulated waveform output of the edge detection procedure utilizing the Sobel operator is shown in Fig. 4.1.5. It displays the computation of gradient magnitude and edge detection in the picture.

4.2 PERFORMANCE ANALYSIS WITH EXISTING HARDWARE IMPLEMENTATION

An investigation of the performance of current hardware implementations of the Sobel edge detection algorithm yields important information about how well resources are used. Though consumption in some areas is slightly higher, overall functionality and resource management are improved in the proposed system when compared. The design of the proposed system integrates sophisticated algorithms or supplementary functionalities to improve edge detection precision and fulfill particular application demands, hence rationalizing the augmented resource distribution.

Resources	**Pujare et.al [7]**	**Santanu et.al [9]**	**Proposed system**
LUT	1%	38	3.61%
FF	1%	38	0.25%
BRAM	36%	-	0.36%
DSP	-	-	0.91%
IO	13%	58	11.50%
BUFIO/BUFG	6%	-	3.31%

Table.4.2.1 Existing System Performance

A comparison of the resource use of the proposed system and the current hardware implementation of the Sobel edge detection algorithm is given in table 4.2.1. Look-Up Tables (LUT), Flip-Flops (FF), Block RAM (BRAM), Digital Signal Processor (DSP), Input/Output (IO), and Clock Buffers (BUFIO/BUFG) are among the important metrics included in the analysis.

Using Verilog for the circular hough transform will be a significant advancement for the project in the future. Better localization results are obtained by accurately identifying the iris's circular boundaries with the use of this method. Furthermore, the Zynq UltraScale+ FPGA board is a good choice for iris recognition workloads that value accuracy and speed due to its real-time data processing capabilities.

5. CONCLUSIONS

The goal of the research is to create an effective edge detection algorithm specifically designed for biometric verification applications by utilizing HDL blocks found in Vitis Model Composer and Simulink. The main goal is to improve edge recognition performance for both grayscale and filtered, colored pictures that are frequently seen in real-world situations. The project increases efficiency and streamlines implementation for deployment on FPGA-based hardware platforms, like the Zynq UltraScale+ MPSoCs, by using HDL blocks integrated with Simulink. The MATLAB graphical environment offered by Simulink's HDL blocks makes it possible to model and simulate hardware parts and algorithms. This integration makes it possible to describe intricate edge detection methods in a high-level graphical environment, which speeds up the development and prototyping process.

In addition to Simulink, Vitis Model Composer facilitates the deployment of developed models onto FPGA hardware by converting them into synthesizable HDL. For biometric verification systems, where precise border delineation is necessary for recognition accuracy, efficient edge detection is critical. In particular, the study applies the Sobel edge detection technique, which is well-known for finding edges and detecting intensity gradients in images. Using the high-performance processing capabilities of the Zynq UltraScale+ MPSoCs, this method is implemented through the Xilinx Vivado IDE. The Sobel edge detection algorithm's remarkable 1000 millisecond execution time when processing a 512x512 resolution image is one of the project's noteworthy accomplishments. This performance shows how effective and appropriate FPGA-based implementations may be for real-time image processing tasks, especially when it comes to biometric verification applications where accuracy and speed are critical.

ACKNOWLEDGMENT

This project is part of the "VLSI Implementation of CNN Architecture for Real-Time Application" project, which is supported by the Indian government and falls under the MeitY "Chips to Startup" (C2S) initiative.

REFERENCES

[1] C. Pradabpet et al. (2009) "An efficient filter structure for multiplier-less Sobel edge detection," in Proceedings of the IEEE International Conference on Signal Processing Systems (ICSPS), pp. 378-382.

[2] C.Perra et al. (2005) "Image Blockiness Evaluation Based Sobel Operator," EURASIP Journal on Applied Signal Processing, vol. 2005, no. 16, pp. 2566-2579.

[3] D. Alghurair, S. S. AI-Rawi. (2013) "Design of Sobel Operator using Field Programmable Gate Array," International Journal of Computer Applications, vol. 68, no. 15, pp. 1-5.

[4] Fernando Martinez Vallina, Christian Kohn, and Pallav Joshi. (2012) "Zynq All Programmable SoC Sobel Filter

Implementation using the Vivado HLS Tool," in Proceedings of the International Conference on Field-Programmable Technology (FPT), pp. 143-146.

[5] I.Yasri, N.H.Hamid, V.V.Yap. (2008) "Performance Analysis of FPGA Based Sobel Edge Detection Operator," International Journal of Computer Science and Network Security, vol. 8, no. 12, pp. 120-125.

[6] Jamie Schiel, Andrew Bainbridge-Smith. (2015) "Efficient Edge Detection on Low-Cost FPGAs," in Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT), pp. 1-6.

[7] Pujare, A., Sawant, P., Sharma, H., & Pichhode, K. (2020). Hardware Implementation of Sobel Edge Detection Algorithm. ITM Web of Conferences, 32, 03051.

[8] K. Wong, A. Chekima, J. Dargham and G. Sainarayanan. (2008) "Palmprint identification using Sobel operator," Pattern Recognition, vol. 41, no. 1, pp. 118-126.

[9] SantanuHalder et al. (2012) "A fast FPGA based architecture for Sobel edge detection," International Journal of VLSI design & Communication Systems (VLSICS), vol. 3, no. 2, pp. 203-214.

[10] Tanvir A Abbasi, MohdAbbasi. (2007) "A proposed FPGA based architecture for Sobel edge detection operator," International Journal of Computer Science and Network Security, vol. 7, no. 8, pp. 186-190.