iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VI **Month of publication:** June 2025

DOI: https://doi.org/10.22214/ijraset.2025.72009

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VI June 2025- Available at www.ijraset.com

Designing SOBEL Edge Detection Using VLSI on FPGA

A. Vani[1], D SathyaNarayana[2], G Anirudh[3], Y Nikhil[4]
[1]*Assistant Professor, Electronics and Communication Engineering, Chaitanya Bharathi Institute of Technology*
[2, 3, 4]*Student, Electronics and Communication Engineering, Chaitanya Bharathi Institute of Technology*

Abstract: ***Edge detection is a critical operation in image processing, widely used in fields such as computer vision, robotics, medical imaging, and object recognition. The Sobel operator, known for its simplicity and effectiveness, computes the gradient of pixel intensities to identify edges within an image. Traditional software-based implementations, while functional, often struggle with real-time processing requirements. The Sobel algorithm is implemented in Verilog HDL, applying 3×3 convolution kernels to compute both horizontal and vertical gradients. These gradients are combined to produce edge magnitudes that highlight the boundaries within the image. The FPGA implementation is developed and tested using the Xilinx Vivado Design Suite. The design is simulated and verified for functional correctness, with results compared to a software-based Python implementation.***
Keywords: SOBEL, FPGA, EDGE.

I. INTRODUCTION

In the field of image processing, edge detection is a critical technique used to identify object boundaries within images. It is fundamental to various applications, including computer vision, robotics, medical imaging, and autonomous systems. Real-time image analysis is increasingly important, driving the need for high-performance computational solutions. Traditional software-based edge detection methods, while effective, often struggle to provide the necessary speed for processing large and high resolution images. This work explores a hardware-based approach to implement the Sobel edge detection algorithm on an FPGA. By utilizing the parallel processing capabilities of FPGAs, the system is designed to achieve high-speed image processing with reduced latency, making it suitable for real-time applications. The Sobel operator, which detects edges by calculating the gradient of an image in both the horizontal and vertical directions, is implemented using the Verilog hardware description language. This project highlights the advantages of using hardware accelerators like FPGAs to improve the performance of image processing tasks.

With the growing use of embedded systems in real - time applications, there is an increasing demand for energy - efficient and rapid computation. This work demonstrates the potential of FPGAs to accelerate image processing tasks, showing how hardware-based solutions can overcome the limitations of conventional software methods. The implementation of Sobel edge detection on an FPGA not only optimizes processing speed but also lays the groundwork for future advancements in hardware-based image analysis systems.

II. RELATED WORK

Gonzalez and Woods (2002) in their book “Digital Image Processing” described the Sobel operator as a fundamental method for edge detection using gradient estimation.[1] Although efficient for software implementations, the method is computationally intensive for large images or real-time processing.

R. C. Gonzalez et al. (2004) proposed enhancements to traditional edge detection techniques but noted the limitation of software-based approaches in real-time scenarios, particularly on low-power embedded systems.[2]

Rohith and Ramya (2015) implemented the Sobel operator on FPGA using VHDL [3]. Their work showed a significant improvement in processing time compared to MATLAB implementations. However, their design lacked optimization for area and power.

S. Mukherjee et al. (2018) worked on real-time edge detection using Verilog on Spartan-6 FPGA [4]. Their design used pipelined architecture to reduce latency and demonstrated successful real-time processing on video frames.

Anitha and Lavanya (2020) presented a comparative study of different edge detection operators on FPGA, concluding that the Sobel operator offers a balanced trade-off between edge quality and hardware complexity [5].

Y. Zhang et al. (2021) demonstrated an energy-efficient implementation of Sobel filtering on FPGA using custom VLSI blocks [6]. Their work emphasized low power design, making it suitable for portable embedded vision systems.

III. METHODOLOGY AND SYSTEM IMPLEMENTATION

A. Methodology

The implementation of the "Sobel Edge Detection Algorithm on FPGA using VLSI design" methodology involves multiple stages, ranging from algorithm understanding to HDL (Hardware Description Language) coding, simulation, synthesis, and testing on FPGA hardware. This section outlines the systematic approach followed for the design and development of the work.

1) Understanding the Sobel Algorithm

The Sobel operator is a discrete differentiation operator used to compute an approximation of the gradient of the image intensity function. It detects edges by calculating the gradient in the x and y directions using two 3x3 convolution kernels:

- Gx (Horizontal kernel):

```
-1  0  +1
-2  0  +2
-1  0  +1
```

- Gy (Vertical kernel):

```
-1  -2  -1
 0   0   0
+1  +2  +1
```

2) Image Preprocessing

Since FPGAs cannot process standard image formats like JPEG or PNG directly, the input image must be converted into a raw pixel intensity format. The following steps are performed:
The image is first converted to grayscale using Python.
Each pixel intensity (0–255) is stored in a 2D matrix.
This matrix is flattened and written into a text file (input.txt), where each value represents one pixel.
This text file is used as an input to the Verilog testbench to simulate image pixel input.

3) HDL Design (Verilog Implementation) :

The core Sobel algorithm is implemented using Verilog HDL. The hardware design is broken down into the following modules:
a. Input Module:
Reads pixel values from a memory or file.
Simulates input image scanning row by row, pixel by pixel.

b. Line Buffering and Windowing Module:
Implements a sliding 3x3 window over the image.
Uses shift registers or RAM blocks to store three lines of image data.
Efficient data reuse and continuous stream processing are ensured.

c. Sobel Convolution Module :
Multiplies the 3x3 window pixels with Gx and Gy kernels.
Adds the products to get Gx and Gy.
Uses absolute value and addition logic to approximate edge magnitude.

d. Thresholding Module :
Compares the edge magnitude with a predefined threshold.
Outputs a binary result (edge = 1, non-edge = 0).

e. Output Module:
Stores or displays the resulting edge image matrix.
Writes results to memory or text file for analysis and comparison.

B. Simulation and Verification

Simulation is performed using *Vivado* or *ModelSim* to verify the functional correctness of the Verilog code. The following steps are followed:
Test bench simulates pixel input from the input.txt file.
Output results are observed in waveform and compared against expected Sobel-filtered values.
Corner and edge cases are tested (e.g., black borders, white noise, high-frequency content).

C. Synthesis and FPGA Implementation

Once verified, the Verilog design is synthesized and implemented on an FPGA board. The synthesis process includes:
Synthesis: Converts HDL code to gate-level netlist using Vivado.
Implementation: Maps the design onto the FPGA's logic elements.
Bitstream Generation: Produces the .bit file to be loaded onto the FPGA.
Programming the FPGA: The bitstream is loaded to the board (e.g., Xilinx Spartan-6 or Artix-7).

D. Software used

Vivado Design Suite (for Verilog synthesis and simulation)
Python (for image preprocessing and result verification)
GTKWave or Vivado Simulator (for waveform analysis)

IV. RESULTS AND DISCUSSIONS

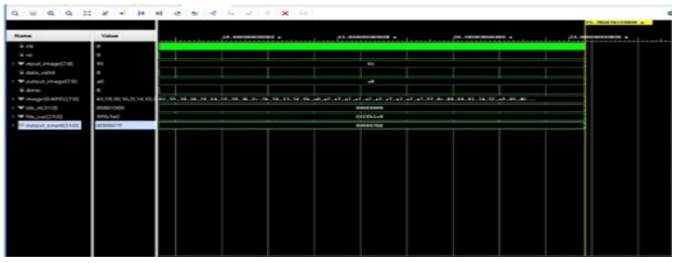
The goal of this work is to implement the Sobel edge detection algorithm on an FPGA using VLSI techniques to achieve efficient real-time edge detection suitable for image processing applications. After completing the design, simulation, synthesis, and hardware implementation stages, evaluated the system for performance, correctness, and efficiency. This section presents the results obtained and discusses their implications in detail.

Simulation Results:

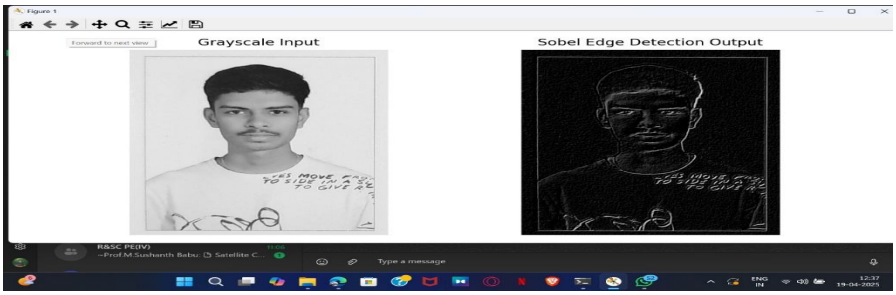
Before implementation on FPGA hardware, the Verilog design was thoroughly tested using simulation tools such as Vivado Simulator and GTKWave. The following outcomes were observed:

- Correct Gradient Calculation: The simulated output waveforms correctly showed the application of the Sobel operator using the Gx and Gy kernels across a 3x3 pixel window.
- Edge Magnitude Output:The output values represented the gradient magnitude approximated by |Gx| + |Gy|, as designed.
- Pipeline Flow: The simulation verified that pipelining was functioning correctly, allowing multiple pixel values to be processed simultaneously in different pipeline stages.

OUTPUTS:WAVE FORM



OUTPUT IMAGES



This figure demonstrates the application of the Sobel edge detection algorithm on a grayscale image of a human subject. The left panel displays the original grayscale image, while the right panel shows the output of the Sobel operator. Prominent edges such as facial contours, hairline, and text on the T-shirt are effectively highlighted, indicating successful edge extraction. This result validates the performance of the algorithm in preserving important structural features from real-world input images.

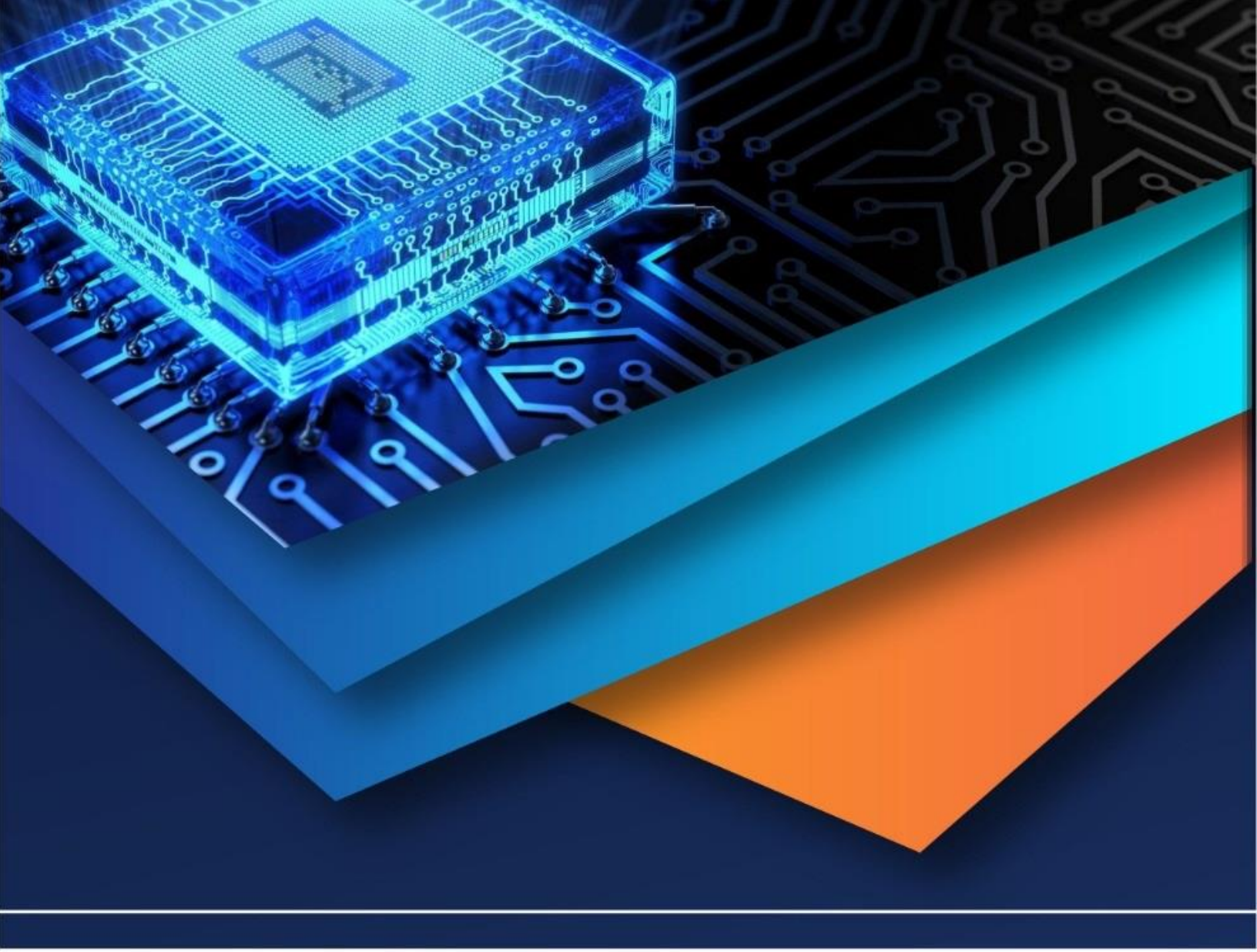
This figure showcases the Sobel edge detection applied to the commonly used Lena image in image processing research. The left panel represents the original grayscale input, and the right panel displays the resulting edge-detected image. The output illustrates clear detection of facial features, hat boundaries, and textured areas, confirming the algorithm's effectiveness in identifying both sharp and soft transitions in intensity. The visual result serves as a benchmark for evaluating edge detection accuracy.

V. CONCLUSIONS

The Sobel edge detection algorithm, which uses horizontal and vertical gradient operators (Gx and Gy), was translated into a hardware description using Verilog HDL. The design was simulated and synthesized using Xilinx Vivado, targeting a suitable FPGA such as Artix-7 or Spartan series and successfully developed a modular, pipelined architecture that reads pixel values from memory, applies convolution using Sobel masks, computes the gradient magnitude, and outputs the resulting edge image. The implementation was tested with image data converted into grayscale matrix format, and the results were validated by comparing with Python-generated output using OpenCV.

REFERENCES

[1] Gonzalez and Woods (2002) in their book "Digital Image Processing" described the Sobel operator as a fundamental method for edge detection using gradient estimation. Although efficient for software implementations, the method is computationally intensive for large images or real-time processing.
[2] R. C. Gonzalez et al. (2004) proposed enhancements to traditional edge detection techniques but noted the limitation of software-based approaches in real-time scenarios, particularly on low-power embedded systems.
[3] Rohith and Ramya (2015) implemented the Sobel operator on FPGA using VHDL. Their work showed a significant improvement in processing time compared to MATLAB implementations. However, their design lacked optimization for area and power.
[4] S. Mukherjee et al. (2018) worked on real-time edge detection using Verilog on Spartan-6 FPGA. Their design used pipelined architecture to reduce latency and demonstrated successful real-time processing on video frames.
[5] Anitha and Lavanya (2020) presented a comparative study of different edge detection operators on FPGA, concluding that the Sobel operator offers a balanced trade-off between edge quality and hardware complexity.
[6] Y. Zhang et al. (2021) demonstrated an energy-efficient implementation of Sobel filtering on FPGA using custom VLSI blocks. Their work emphasized low power design, making it suitable for portable embedded vision systems.
[7] A. G. Dandapat, S. Mukhopadhyay, and B. N. Subudhi, "FPGA Implementation of Sobel Edge Detection Algorithm," International Journal of Computer Applications, vol. 64, no. 2, pp. 1–6, 2013.
[8] R. Singh, D. Singh, and S. S. Ahuja, "Real-Time Image Edge Detection Using FPGA Implementation," International Journal of VLSI Design & Communication Systems, vol. 4, no. 3, pp. 23–32, 2013.
[9] T. J. Todman, G. A. Constantinides, and W. Luk, "Reconfigurable computing: Architectures and design methods," IEE Proceedings - Computers and Digital Techniques, vol. 152, no. 2, pp. 193–207, 2005.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY