

FPGA-Based Image EDGE Detection

Tejas Nanasaheb Rahane¹, Sakshi Vikram Shelke², Shravani Sambhaji Hinge³

Department of Artificial Intelligence and Data Science, Sengunthar Engineering College, Tiruchengode, Tamil Nadu, India

Abstract- Edge detection is an important operation in digital image processing because it identifies boundaries where image intensity changes sharply. These boundaries provide useful information for object detection, shape analysis, machine vision and robotics. This paper presents a proposed FPGA-based image edge detection system using the Sobel operator. The input image is acquired from a camera or stored image source and is converted to a suitable grayscale pixel stream. A 3×3 sliding window is formed with line buffers and registers. The Sobel horizontal and vertical kernels calculate gradient components, which are combined and compared with a threshold to generate an edge image. FPGA parallelism and reconfigurable logic can support high-throughput, low-latency processing for suitable designs. The design can be described using Verilog HDL or VHDL and developed with tools such as AMD Xilinx Vivado or Intel Quartus Prime. Performance must be verified on the selected board using measured timing, resource utilization and image-quality results.

Index Terms— FPGA, Image Processing, Edge Detection, Sobel Operator, Verilog HDL, Real-Time Vision.

I. INTRODUCTION

Digital image processing is an important technology used to analyze images and extract useful information from them. Nowadays, it is widely used in various fields, such as medical imaging, robotics, industrial automation, security systems, traffic monitoring, and autonomous vehicles. It helps computers understand the contents of an image by identifying important features, shapes, and objects. One of the basic and important operations in digital image processing is edge detection, which helps identify the boundaries of objects in an image.

Edge detection is a technique used to find sudden changes in brightness or intensity between neighbouring pixels. These changes usually represent the boundaries of different objects, lines, or surfaces. For example, when an image of a vehicle is processed, edge detection can help identify the outline of the vehicle and distinguish it from the surrounding background. Similarly, it can be used to identify shapes in industrial products, detect objects in robotic systems, and extract important features

from medical images. Therefore, edge detection plays an important role in many computer vision applications.

Traditional image processing techniques are commonly implemented using software on computers and processors. Although these methods are useful, processing a large number of image pixels can take considerable time, especially when images or videos must be processed continuously. Applications

such as robotics, industrial inspection, and real-time monitoring require fast and reliable processing. To meet these requirements, hardware-based image processing can be used. One suitable technology for this purpose is the Field-Programmable Gate Array, commonly known as an FPGA.

An FPGA is a programmable digital integrated circuit that allows users to design and implement specific hardware functions according to their requirements. It contains configurable logic blocks, programmable interconnections, and other hardware resources that can be arranged to perform different operations. One of the main advantages of an FPGA is its ability to perform multiple operations in parallel. This feature makes it useful for applications that require high processing speed and low latency. FPGAs are also reconfigurable, which means that their hardware functionality can be modified to support different algorithms or design requirements.

The project titled “FPGA-Based Image Edge Detection” focuses on implementing an image edge detection algorithm using FPGA hardware. The main purpose of this project is to identify important boundaries in an image through a digital hardware circuit. The proposed system uses the Sobel operator, which is a commonly used method for detecting edges in digital images. The Sobel operator is suitable for hardware implementation because its calculations mainly involve addition, subtraction, and simple arithmetic operations. These operations can be efficiently arranged in a digital circuit.

The Sobel operator uses two 3×3 convolution kernels to examine the intensity values of neighbouring pixels. One kernel calculates the intensity changes in the horizontal direction, while the other calculates the changes in the vertical direction. These calculations generate two gradient values, known as G_x and G_y . The gradient values are combined to estimate the strength of the edge at each pixel. A threshold value is then applied to distinguish edge pixels from non-edge pixels. The final output is an edge-detected image in which the important boundaries of objects become more visible.

In the proposed system, an input image is supplied from a camera or a stored image source. If the image is coloured, it can first be converted into grayscale to simplify the processing. The pixel data is then transferred to the FPGA processing unit. Since the Sobel operator requires nine neighbouring pixels for each calculation, line buffers and registers can be used to create a 3×3 pixel window. The FPGA performs the required gradient calculations and threshold comparison to generate the final edge image. The output can be stored in memory or displayed using a suitable output interface.

The hardware design can be developed using a Hardware Description Language, such as Verilog HDL or VHDL. Software tools such as AMD Xilinx Vivado or Intel Quartus Prime can be used for writing, simulating, synthesizing, and implementing the design. MATLAB or Python can also be used to develop a reference implementation of the Sobel algorithm and compare its output with the FPGA results. Simulation helps verify the correct operation of the design before it is programmed onto an FPGA development board.

The main aim of this project is to understand how image processing algorithms can be implemented using digital hardware. It provides practical knowledge of FPGA architecture, digital logic design, Verilog HDL programming, and image processing techniques. By implementing the Sobel operator on an FPGA, the project demonstrates how parallel processing can support efficient image analysis. The actual processing speed, hardware resource utilization, and output quality can be evaluated through simulation and testing on the selected FPGA board.

FPGA-based edge detection has potential applications in robotics, industrial quality inspection, surveillance systems, medical image processing, traffic monitoring, and autonomous systems. It can be used as an initial processing stage before

more advanced operations, such as object recognition and feature extraction. In the future, the proposed system can be improved by incorporating noise reduction, adjustable threshold values, higher-resolution image interfaces, and advanced image processing algorithms. Overall, FPGA-Based Image Edge Detection provides a practical foundation for understanding and developing hardware-accelerated image processing systems.

II. WORKING

The working of the FPGA-Based Image Edge Detection system begins with providing an input image to the system. The image can be obtained from a camera or loaded from previously stored image data. The main purpose of the system is to identify the boundaries of objects present in the image by detecting sudden changes in pixel intensity. Before edge detection, the input image is checked and prepared for further processing. If the input image is coloured, it can be converted into a grayscale image to simplify the calculations. In a grayscale image, each pixel represents the brightness level of that particular point, making it easier to identify changes in intensity between neighbouring pixels.

After pre-processing, the image pixel data is transferred to the FPGA processing unit. The FPGA is programmed to perform the edge detection algorithm using Verilog HDL or VHDL. The Sobel operator is used in this project to identify intensity changes in horizontal and vertical directions. Since the Sobel algorithm requires information from nine neighbouring pixels, the FPGA uses line buffers and registers to store the required pixel values and create a 3×3 sliding window. This window moves across the image pixel by pixel, allowing the system to examine different regions of the image and calculate the corresponding edge information.

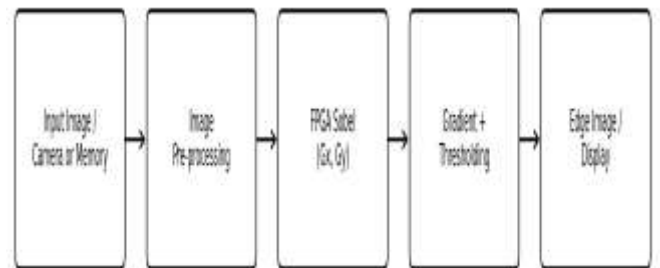
The Sobel operator consists of two 3×3 convolution kernels, called G_x and G_y . The G_x kernel detects intensity changes in one direction, while the G_y kernel detects changes in the perpendicular direction. Each kernel performs mathematical calculations on the pixel values within the 3×3 window. The resulting values represent the horizontal and vertical gradients of the image. A larger gradient value indicates a stronger change in intensity and may represent an important object boundary. These calculations are performed by the FPGA using its digital logic resources, allowing the required arithmetic operations to be organized efficiently in hardware.

Once the horizontal and vertical gradient values are calculated, the system determines the strength of the edge at the current pixel. The gradient magnitude can be calculated using the square root of the sum of the squares of G_x and G_y . However, to simplify hardware implementation, an approximate method such as adding the absolute values of the two gradients can also be used. The selected method depends on the design requirements and available FPGA resources. The calculated edge strength is then compared with a predefined threshold value. If the edge strength is greater than the threshold, the pixel is classified as an edge pixel. Otherwise, it is classified as a non-edge pixel. This process helps separate important boundaries from relatively uniform regions of the image.

After thresholding, the FPGA generates the final edge-detected image. In a typical binary output, edge pixels are represented by white pixels, while non-edge pixels are represented by black pixels. As a result, the boundaries of objects become clearly visible against the background. The processed image can then be transferred to an output display, stored in memory, or sent to another processing unit for further analysis. For example, the detected edges can be used as input for object detection, shape identification, or other computer vision operations.

The FPGA can process image pixels through a series of hardware stages, with different operations working on different pixels at the same time. This technique is known as pipelining and can improve the overall processing throughput. However, the actual performance depends on the FPGA board, clock frequency, memory arrangement, image resolution, and hardware design. The system should be verified using simulation before being implemented on the development board. Tools such as AMD Xilinx Vivado or Intel Quartus Prime can be used to simulate the design, check its hardware resource utilization, and analyze its timing performance.

Overall, the working principle of the proposed system involves image acquisition, pre-processing, pixel-window generation, Sobel gradient calculation, thresholding, and output generation. Each stage performs a specific operation and contributes to the final edge-detected image. By implementing these stages using FPGA hardware, the project demonstrates how a digital image processing algorithm can be converted into a practical hardware design. The system can be further improved by adding noise reduction, adjustable threshold values, camera interfacing, and more advanced edge detection techniques.



Advantages:

- Parallel hardware operations can provide high throughput for continuous image streams.
- Pipelining can reduce the time between successive processed pixels.
- The FPGA logic can be reconfigured to adjust the algorithm or threshold.
- A dedicated implementation can offer predictable processing latency.
- The design can be integrated with cameras, processors, memory and display interfaces.

Disadvantages:

- Initial FPGA development requires knowledge of digital design, HDL coding and timing constraints.
- The development board and camera/display interfaces may increase prototype cost.
- Line buffers and external memory may be needed for larger image sizes or higher resolutions.
- Image quality depends on noise, threshold selection, pixel precision and border handling.
- Actual frame rate, resource usage and power consumption depend on the selected FPGA and implementation; these must be measured rather than assumed.

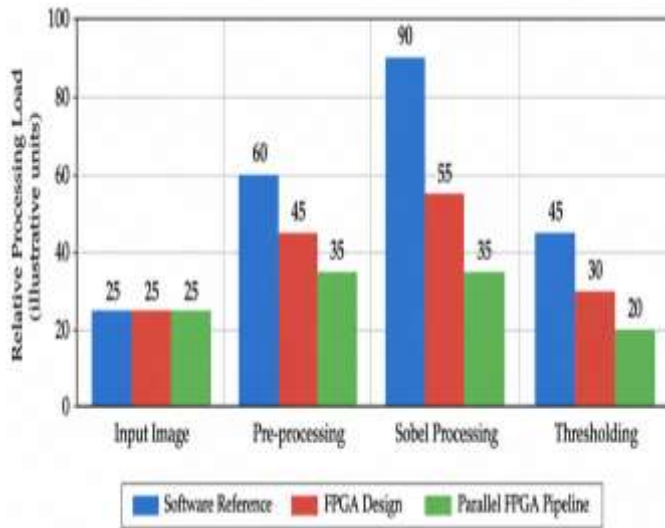


Chart -1: Comparative Processing Performance of FPGA-Based Image Edge Detection.



Figure 1: FPGA-Based Image Edge Detection System Using the Sobel Operator

Table -1: Component List

Sr. No.	Component / Tool	Quantity	Purpose
1	FPGA development board	1	Runs the edge-detection hardware
2	Input image source / camera	1	Supplies pixel data
3	Image memory or line buffers	As required	Stores image rows and pixel windows
4	Display or output interface	Optional	Shows or exports the edge image
5	USB/JTAG programming connection	1	Loads the FPGA design
6	Vivado or Quartus Prime	1 software tool	Synthesis, implementation and timing analysis
7	Verilog HDL or VHDL	—	Hardware description
8	MATLAB or Python	Optional	Reference algorithm and output comparison

III. CONCLUSION

FPGA-based image edge detection system using the Sobel operator. The architecture includes image input, optional pre-processing, 3×3 window generation, gradient calculation and thresholding. The parallel and reconfigurable nature of FPGA hardware makes it a useful platform for vision tasks that need continuous processing. The next step is to implement the design in Verilog HDL or VHDL, verify it through simulation, and test it on a selected FPGA board. Future improvements may include noise filtering, configurable thresholds, higher-resolution streaming interfaces, and integration with machine-learning-based vision systems

REFERENCES

1. S. Ravichandran, H.-K. Su, W.-K. Kuo, D. Dhanasekaran, M. Mahalingam, and J.-P. Yang, "Parallel Processing of Sobel Edge Detection on FPGA: Enhancing Real-Time Image Analysis," Sensors, vol. 25, no. 12, Art. no. 3649, 2025, doi: 10.3390/s25123649.
2. Y. An, Q. Yuan, and H. Zhang, "Edge Detection Using Sobel Algorithm and YCbCr Colour Space

- Optimized on FPGA,” *Informatica*, vol. 49, no. 7, 2025, doi: 10.31449/inf.v49i7.6998.
3. “Implementation of Improved Sobel Edge Detection on FPGA,” in *Proc. 9th Int. Conf. Computational System and Information Technology for Sustainable Solutions (CSITSS)*, 2025, doi: 10.1109/CSITSS67709.2025.11294068.
 4. G. Sujatha, A. Sushitha, V. Y. Datta, K. Shashidhar, M. PavanKumar, and B. Kiranmayee, “FPGA Implementation for Image Improved Edge Detection Algorithm Using Xilinx ISE Simulator,” in *Proc. 4th Int. Conf. Information Technology, Civil Innovation, Science, and Management (ICITSM)*, 2025, doi: 10.4108/eai.28-4-2025.2358041.
 5. “Efficient Hardware Implementation of the Sobel Operator using FPGA,” in *Proc. IEEE Int. Conf. Intelligent Signal Processing and Effective Communication Technologies (INSPECT)*, 2025, doi: 10.1109/INSPECT67393.2025.11350865.
 6. V. S. S., A. V. R., and N. M., “Implementation of Modified SOBEL Edge Detection for Brain Tumor Detection in MRI Using FPGA,” in *Proc. Int. Conf. Networks & Advances in Computational Technologies (NetACT)*, 2025, pp. 1–6, doi: 10.1109/NetACT65906.2025.11188401.
 7. A. Poddar, P. Verma, and S. Singh, “Design & Implementation of Sobel Edge Detection via Customized MAC IP in Vivado,” in *Proc. 3rd World Conf. Communication & Computing (WCONF)*, 2025, pp. 1–6, doi: 10.1109/WCONF64849.2025.11233433.
 8. K. Navinkumar, R. Logesh, P. VishnuBabu, and A. V. Ananthalakshmi, “FPGA Implementation of Sobel Edge Detection Algorithm,” *EAI Endorsed Transactions on Internet of Things*, vol. 10, no. 1, 2024, doi: 10.4108/eetiot.5148.
 9. Y. Liu, F. Huo, and J. Zhang, “Design and Implementation of an FPGA-Based Real-Time Edge Detection System Using an Enhanced Sobel Operator,” in *Proc. 6th Int. Conf. Electronics, Circuits and Information Engineering (ECIE)*, 2026, pp. 211–216, doi: 10.1109/ECIE69549.2026.11565630.
 10. C. Peddini and S. Sruthi, “FPGA-Based Real-Time Image Edge Detection Using Pipelined Sobel and Canny Operations,” *International Journal of Engineering & Extended Technologies Research*, 2026, doi: 10.15662/IJEETR.2026.0802080.